НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

На правах рукопису
УДК 004.896

«До захисту допущено»
В.О. завідувача кафедри ММСА
_____ О.Л. Тимощук
«___»_____2020 р.

Магістерська дисертація

на здобуття ступеня магістра за спеціальністю 122 Комп’ютерні науки

на тему: «Побудова інформаційної інфраструктури міста за допомогою рекомендаційних систем»

Виконала:
студентка ІІ курсу, групи КА-93мп
Мозолевська Марія Олегівна _____

Науковий керівник:
Доцент кафедри ММСА,
к.т.н., доц. Дідковська М.В.

_____

Рецензент:
Доцент кафедри програмного забезпечення
комп'ютерних систем
КПІ ім. Ігоря Сікорського,
к. т. н., доцент. Заболотня Т.М

_____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ
2020

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Рівень вищої освіти – другий (магістерський)

Спеціальність (спеціалізація) — 122 «Комп'ютерні науки» («Системи штучного інтелекту»)

ЗАТВЕРДЖУЮ

В.О.Завідувача кафедри

_____ О.Л. Тимощук

«___»_____20__ р.

ЗАВДАННЯ

на магістерську дисертацію студенту Мозолевській Марії Олегівні

1. Тема роботи «Побудова інформаційної інфраструктури міста за допомогою рекомендаційних систем», керівник роботи Дідковська Марина Віталіївна, к.т.н, доцент, затверджені наказом по університету від «02»листопада 2020р. № 3182-с

2. Термін подання студентом дисертації:_____

3. Об'єкт дослідження: інформаційна інфраструктура «розумного» міста для допомоги жителям у організації соціально-культурного життя

4. Предмет дослідження: методи побудови систем рекомендації для побудови інформаційної інфраструктури «розумного» міста

5. Перелік завдань, що потрібно розробити:

1) дослідити наукові дослідження у сфері побудови інформаційної інфраструктури міста;

2) провести аналіз існуючих методів побудови систем рекомендації;

3) виконати проектування системи рекомендації;

4) реалізувати один з модулів спроектованої системи;

5) розробити стартап-проект виведення на ринок результатів дослідження;

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:

1) Різновиди систем рекомендацій;

2) Етапи побудови рекомендаційної системи;

3) Логіка роботи рекомендаційних систем.

8. Дата видачі завдання 01.09.2020

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів роботи
1.	Концептуальний вступ дисертації. Формулювання об'єкта, предмета, цілі, завдань, новизни, практичної значущості результатів	04.09.2020-20.09.2020
2.	Перший розділ. Дослідження предметної області	21.09.2020-29.09.2020
3.	Другий розділ. Методи побудови рекомендаційної системи для розробки інформаційної інфраструктури розумного міста	30.09.2020-08.10.2020
4.	Третій розділ. Розробка системи рекомендації	19.10.2020-19.10.2020
5.	Четвертий розділ. Розроблення стартап-проекту	20.10.2020-24.11.2020
6.	Концептуальні висновки. Перспективи розвитку отриманих рішень	25.11.2020-30.11.2020

Студент М.О. Мозолевська

Науковий керівник дисертації М.В. Дідковська

РЕФЕРАТ

Магістерська дисертація: 108с., 36 рис., 17 табл., 26 джерел.

Мета роботи – розробка системи рекомендації для побудови інформаційної інфраструктури розумного міста на основі датасету, побудованого на опитуванні жителів міста та їх переваг в організації соціально-культурного життя. Об’єктом дослідження є інформаційна інфраструктура міста для допомоги жителям міста в організації соціально-культурного життя.

Предметом дослідження є методи розробки рекомендаційних систем для побудови інформаційної інфраструктури міста. В роботі досліджується проблема рекомендації жителям розумного міста заходів та інформації, що будуть персоналізованими та враховувати інтереси кожного окремого користувача системи рекомендації шляхом аналізу їх вподобань та побудови математичних моделей на основі історичних даних.

Виконано аналіз досліджень та наукових праць у сфері розробки інформаційної інфраструктури міста, досліджено методи побудови рекомендаційних систем, аналіз методів систем рекомендації, що можна застосувати для розробки додатку, що будуть персоналізовано рекомендувати користувачам-жителям міста заходів. Результатом роботи є аналіз підходів побудови систем рекомендації, розробка персоналізованої системи рекомендації . Під час роботи реалізовано систему з використанням мови програмування Python, що надає широкий спектр бібліотек для обробки та аналізу даних.

СИСТЕМА РЕКОМЕНДАЦІЇ, РОЗУМНЕ МІСТО, КОЛАБОРАТИВНА ФІЛЬТРАЦІЯ, ФІЛЬТРАЦІЯ НА ОСНОВІ ВМІСТУ, ГІБРИДНІ РЕКОМЕНДАЦІЙНІ СИСТЕМИ

ABSTRACT

Master's thesis: 108 pp., 36 fig., 17 table, 26 sources.

The purpose of the work is to develop a system of recommendations for building an information infrastructure of a smart city on the basis of a dataset based on a survey of city residents and their preferences in the organization of socio-cultural life.

The object of the study is the information infrastructure of the city to help city residents in organizing socio-cultural life.

The subject of the research is the methods of developing recommendation systems for building the information infrastructure of the city.

The paper examines the problem of recommending to residents of a smart city activities and information that will be personalized and take into account the interests of each individual user of the recommendation system by analyzing their preferences and building mathematical models based on historical data.

The analysis of researches and scientific works in the field of development of an information infrastructure of the city is executed, methods of construction of recommendation systems, the analysis of methods of systems of the recommendation which can be applied to application development which will be personally recommended to recommend to users-inhabitants of the city are investigated.

The result is an analysis of modern methods and approaches to building recommendation systems, development of a personalized recommendation system. During the work, the system was implemented using the Python programming language, which provides a wide range of libraries for data processing and analysis.

RECOMMENDATION SYSTEM, REASONABLE CITY, COLLABORATIVE FILTRATION, CONTENT-BASED FILTRATION, HYBRID RECOMMENDATION SYSTEMS

ЗМІСТ

ВСТУП

Дослідження та розробки у напрямку побудови інформаційної інфраструктури міста з’явилися як відповідь на стрімкі процеси урбанізації. Розвиток цієї сфери є очікуваним, оскільки більше половини світового населення проживає у містах і очікується, що це число збільшиться до 70% до 2050.

Таким чином, виникло поняття розумного міста (англ. – smart city), метою побудови якого є допомогти мешканцям міст у плануванні свого повсякденного соціального життя в умовах постійної урбанізації шляхом інтеграції рішень на основі інформаційно-комунікаційних технологій (ІКТ). Розумні міста прагнуть використовувати інноваційні ІКТ-рішення для вирішення міських проблем, пов'язаних з навколишнім середовищем, людьми, мобільністю, безпекою, економікою, управлінням ресурсами, охороною здоров'я тощо. Розумне місто інвестує в ІКТ для підтримки сталого соціально-економічного розвитку, поліпшення якості життя та розумне використання природних ресурсів. У побудові розумного міста ІКТ використовується як основний фактор, що сприяє поліпшенню інтеграції даних, які мають бути перетворений у корисну інформацію та інформацію про знання для сталого розвитку міст.

Для розробки інформаційної платформи розумного міста використовують такі ІКТ-технології як: великі дані, інтернет речей, хмарні обчислення тощо. Одним із способів побудови такої платформи є розробка системи рекомендації з використанням машинного навчання. Таким чином, це дослідження інтегрує міркування на основі конкретних випадків для розробки системи рекомендацій щодо сприяння розвитку розумного міста.

Відповідно, для побудови рекомендаційної системи, яка буде служити інформаційною інфраструктурою розумного міста, необхідні тренувальні дані.

Було прийнято рішення використати дані, опубліковані PAVIC як результати опитування , присвячене розумному місту, у якому було зібрано результати опитування 1076 респондентів [4].

Це опитування було повністю анонімним і було спрямоване на покращення життя громадян у майбутньому розумному місті. Таким чином, на основі даних реальних людей можна буде побудувати якісну рекомендаційну систему, яка буде враховувати запити суспільства. Окрім того, очікується, що результати цього дослідження дадуть цінну інформацію для практиків для розробки практичних стратегій у побудові інформаційної платформи розумного міста.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд літератури

Поняття розумного міста було вперше розглянуто в 1994 році, але з 2011 року було опубліковані більш наближені до сьогоднішніх реалій дослідження та практики в області побудови та розвитку розумних міст. У деяких із цих досліджень були опубліковані підходи, що використовуються сучасними спеціалістами для побудови розумних міст. Повний огляд літератури на дану тематику можна розглянути у *Таблиці 1.1* На основі оглянутих досліджень було визначено кілька підходів, що будуть використані у даній магістерській дисертації для побудови інформаційної інфраструктури розумного міста.

Таблиця 1.1 – Огляд літератури про підходи побудови розумного міста

Автори, рік та внесок	**Вирішена проблема**	**Застосований метод дослідження**	**Обмеження**
Бадій та ін. (2017) надав аналіз та оцінку архітектури розумного міста, орієнтованого на знання [2]	Спрямоване на визначення архітектури розумного міста вирішення широкого кола процесів і даних	Порівняння готових рішень для розумних міст для агрегування даних та для інтерфейсу програмування інтелектуального міста (API)	Запропонована архітектура була оцінена в умови обчислювальних та мережевих витрат лише на хмарах
Хан та ін. (2017) впровадили заявку на залучення громадян до знань сприяти прийняттю рішень у розумних містах [3]	Запропонували системну архітектуру, яка є в основі різних інноваційних функцій, доказ концепції функції автоматизованого зворотного зв'язку для розумного міста	Застосований експеримент як доказ концепції на основі 3D-візуалізації та зворотного зв'язку об'єкту для вищезазначеного випадку використання	Автори не досліджували практичність функції зворотного зв’язку на різні теми

Продовження таблиці 1.1

Автори, рік та внесок	**Вирішена проблема**	**Застосований метод дослідження**	**Обмеження**
Морено та ін. (2017) розробили архітектуру розумного міста, заснованого на техніці великих даних [4]	Спрямований на надання прибуткових послуг розумного міста - управління енергоспоживанням, досягнення розумних будівель та досягнення розумний транспорт	Запропоновані два сценарії аналізу великих даних, де один стосується розумного кампуса, а інший - трамвайного сполучення	Дані, зібрані з джерел зондування натовпу не використовувалась для покращення оцінки конфігурації міської мобільності
Гаур та ін. (2015) розробили багаторівневий смарт міську архітектуру на основі семантичної мережі технології та теорії невизначеності Демпстера Шафера [5]	Спрямований на використання збільшеного обсягу даних для ефективного управління та аналізу даних генерувати інформацію, яка може допомогти управління використанням ресурсів у містах	Зібрані неоднорідні дані як CSV, твіти, схеми баз даних і текстові повідомлення в пояснити функціональність архітектури та деякі контекстні сценарії в реальному часі	Аспекти сумісності та масштабованості даних не були визначені в запропонованій архітектурі
Anthopoulos and Vakali (2012) розробили а багаторівнева архітектура цифрового міста [6]	Зосереджено на виділенні та вимірюванні розумних містобудівний взаємозв'язок і визначає місця зустрічі серед них	Дані були зібрані з вторинного джерела	Розвинена архітектура не була емпірично перевіреною

Кінець таблиці 1.1

Автори, рік та внесок	**Вирішена проблема**	**Застосований метод дослідження**	**Обмеження**
Чурабі та ін. (2012) розробили структуру допомогти зрозуміти концепцію розумних міст [7]	Мав на меті вказати важливі фактори, які мають бути прийнятий для ініціатив розумного міста	Концептуальна модель, дані не збиралися	Жодних емпіричних даних не було повідомлено авторами
Дікін (2012) запропонував уніфіковану модель обслуговування електронного уряду інтелектуального міста як розумний постачальник вдосконалених веб-програм послуги [8]	Мав на меті вказати, наскільки розумні міста під керівництвом університетів дослідити можливості, які надає спільнота практик промисловості, щоб стати розумними постачальниками веб-сервіси	Застосована онтологія знань для передачі, використання, застосування, створення знань, видобуток, нарощування потенціалу та знання виробництва	Емпіричні дані не повідомляютьс я дослідником
Девід та ін. (2012) розвинене розумне місто архітектура системи для збору та обміну даними через мобільний пристрій для вирішення проблем пов'язані із розумним містом [9]	Мав на меті практично представляти такий підхід візуалізації зібраних міських даних, які будуть використані міськими користувачами та пов'язаними з ними послугами	Використовував три тематичні дослідження на основі системи міської доставки товарів, динамічної система виділення смуги руху та комунікації	Застосовано загальний підхід, заснований на довідкових послугах та архітектурі лише для даних візуалізації
Нам і Пардо (2011) розробили концепцію smart місто на основі технологій, установ та люди як основні виміри [10]	Мав на меті представити основні фактори розумного міста ініціатива шляхом вивчення різних різноманітних дослідження розумного міста	Концептуальна модель. Дані не збиралися.	Політика інновацій, задіяних у розумному місті управління не було повністю вивчене

1.2 Складові розумного міста

На *Рис. 1.1* показані визначені складові розумного міста та відповідні ініціативи, запропоновані Уашборном Д. [2] Кожен із цих вимірів розглядається нижче.

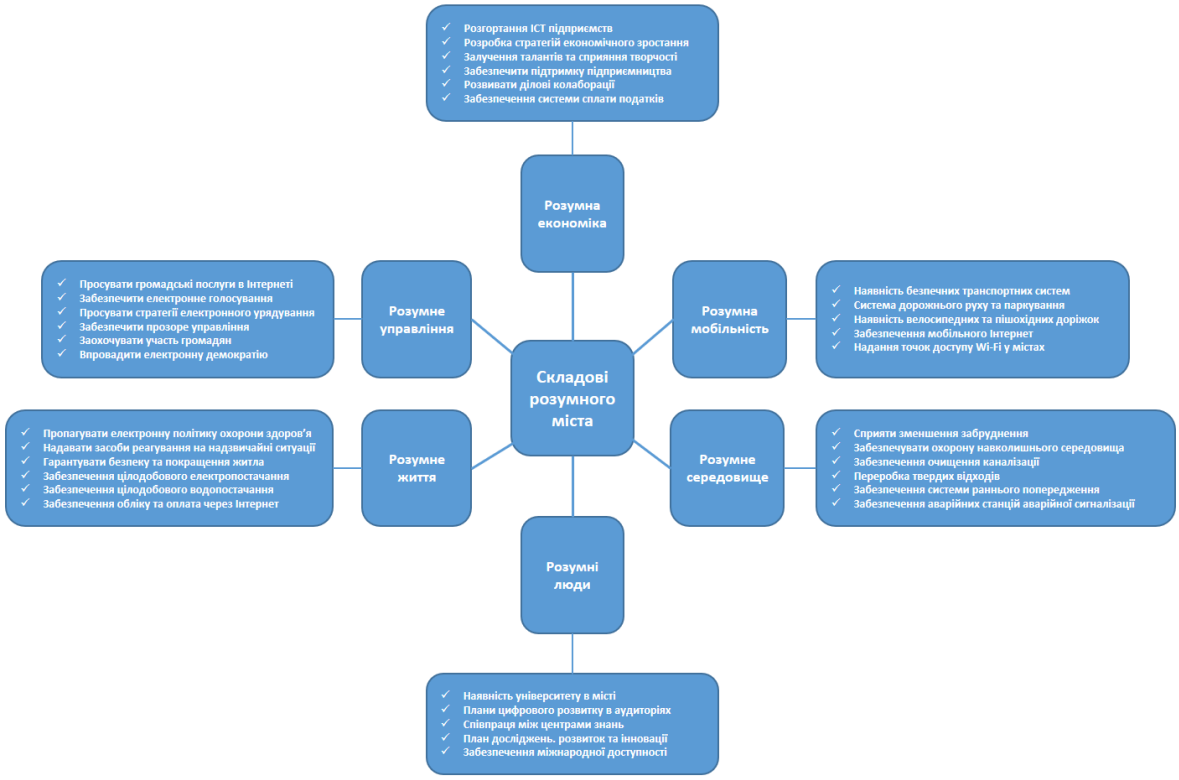


Рисунок 1.1 – Складові розумного міста

Розумна економіка – це вимір, який визначає доцільність робочої сили, інновації, підприємництво, економічний імідж та здатність міста перетворюватися на зелене місто. Вона також включає високу економічну конкурентоспроможність,

що покращує економічний розвиток міста. Розумна економіка в основному зумовлена інтеграцією з глобальними ринками і має здатність вдосконалювати економічну конкурентоспроможність міста. Крім того, здатність міста залучати відвідувачів, бізнес, столицю та таланти також збільшує його економічне зростання.

Розумна мобільність передбачає доступність, а також наявність безпечних систем переїзду, сучасних зручностей та зелені інфраструктури. Це стосується місцевої доступності, наявності безпечної, сучасної та стійкі транспортні системи. Крім того, розумна мобільність передбачає надання громадянам доступу до інноваційних технологій для полегшення міської рутини в містах. Крім того, повинні бути доступними інфраструктури підтримки мобільного доступу до інформації та мобільність міст для транспортних послуг. Отже, розумна мобільність тягне за собою впровадження ІКТ як спосіб активізації транспортних операцій з метою досягти доступної мобільності. Таким чином, міста повинні реалізовувати ІКТ для підвищення мобільності як підхід до побудови цифрових технологій та інтегрованої транспортної мережі.

Розумне середовище. У розумних містах вимір навколишнього середовища пов'язаний зі збереженням природних ресурсів, таких як вода, чисте повітря, земля тощо. Це передбачає екологічне використання природних ресурсів, захист природного середовища, зменшення забруднення та стійке управління ресурсами. Розумне середовище також визначається привабливістю природних умов та відповідальним управлінням ресурсами за постійного зменшення ресурсів та постійно зростаючих вимог. Відповідно, розумне середовище означає використання технологій для захисту та зберігання природного середовища міста та характеризується безпекою та довірою, використанням ІКТ для вдосконалення культурних ініціатив, муніципальної безпека для діджиталізації традиційні активностей.

Розумні люди включають соціальні спільноти та людей, які проживають у місті. Участь та залучення людей – це критерій, який впливає на досягнення розумного міста. Отже, щоб місто було розумним, громадяни також повинні бути розумними, щоб досягти інклюзивного, інноваційного та стійкого міста. Розумні люди – це також соціальний та людський капітал, толерантність, творчість та участь у публічних заходах. У розумному місті мешканці розумні з точки зору своєї кваліфікації та рівня освіти, а також значення соціальної співпраці в умовах інтеграції суспільного життя та їх здатність спілкуватися між собою. За словами Тахіра та Малека [3], можуть впливати й інші фактори, що включають соціальну та етнічну різноманітність, рівень кваліфікації, схильність до життя і відкритість.

Розумне життя передбачає поліпшення якості життя громадян, змінивши житло, громади, робоче місце, енергію та транспортної інфраструктури у зелене середовище. Розумний спосіб життя покращує розуміння того, як технології та суспільство взаємодіють між собою на користь громадян. Отже, розумний життя - це адаптація факторів, що становлять осмислене та щасливе життя. Це передбачає кілька факторів (наприклад, охорона здоров'я, освіта, туризм, безпека, культура тощо), які покращують якість життя мешканців, що ведуть до більш гармонійного і задовільного життя.

Розумне управління містом передбачає використання ІКТ для підтримки прийняття рішень та планування для тих, хто розробляє політику. Це передбачає вдосконалення процедур самоврядування та зміну способу розподілу громадських послуг. Більше того, він стурбований державним керівництвом та послугами для більшої ефективності та постійності розвиток за допомогою ІКТ-інновацій, таких як електронна демократія або електронне урядування. Це передбачає посилення демократичних процедур та перетворення способу надання державних послуг містом лідерів.

1.3 Огляд континууму знань для розумного міста

Розумне місто складається з даних, інформації, знань та мудрість, яка відома як континуум знань як показано на *Рис.1.2.*

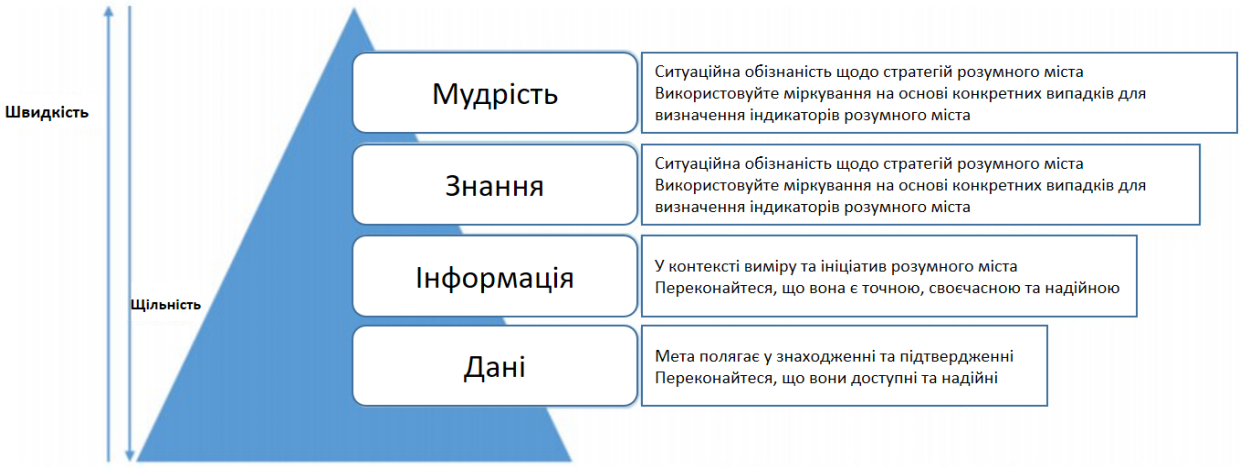


Рисунок 1.2 – Управління знаннями для досягнення розумного міста

Відповідно, **дані** стосуються набору цифр, тексту або символів в неорганізованому або необробленому форматі, що потрібно обробити, щоб отримати результат. Дані складаються з нефільтрованого, непереробленого базового формату.

Інформація включає оброблені дані, які структуровані, як правило, за допомогою комп’ютера, щоб надати такої форми, яку можна інтерпретувати. Коли шляхом обробки даних отримується інформація, вона стає значущою, набуває значення.

І навпаки, **знання** стосуються індивідуального придбання інформації, наприклад розуміння інформації або фактів щодо вирішення певних питань. Знання існують в особистості і трапляються лише тоді, коли розуміння та досвід користувачів є практичними щодо даних та інформації.

Мудрість – це не лише знання, а і здатність діяти в форматі ноу-хау та розуміння причин, які існують у свідомості окремої людини. На *Рис.1.2.* показано трансформацію даних за допомогою управління знаннями для досягнення розумного міста. Крім того, розумні міста базуються на знаннях та креативних схемах, спрямованих на поліпшення екологічних, соціально-економічних та соціальних показників міст.

Відповідно, для планування та керування містом необхідний інноваційний підхід за допомогою інженерії знань, що передбачає розробку та розгортання інфраструктури, що базується на знаннях. Таким чином, на *Рис. 1.2* зображені дані, що представляють необроблені показники, інформація стосується даних зі значенням та знаннями, а інформація може бути корисною для вирішення проблеми. Більше того, застосовуючи знання, можна досягти свого роду прийняття рішень (мудрості) для сприяння розвитку розумного міста.

На *Рис.1.3.* зображена роль знань у просуванні розумних технологій міська поліція. Більше того, *Рис.1.3* ілюструє роль інфраструктури знань у розумному місті, яке базується на фізичному рівні, що включає сенсори, зв’язки та дані про розміри розумного міста та пов'язані з ними ініціативи.

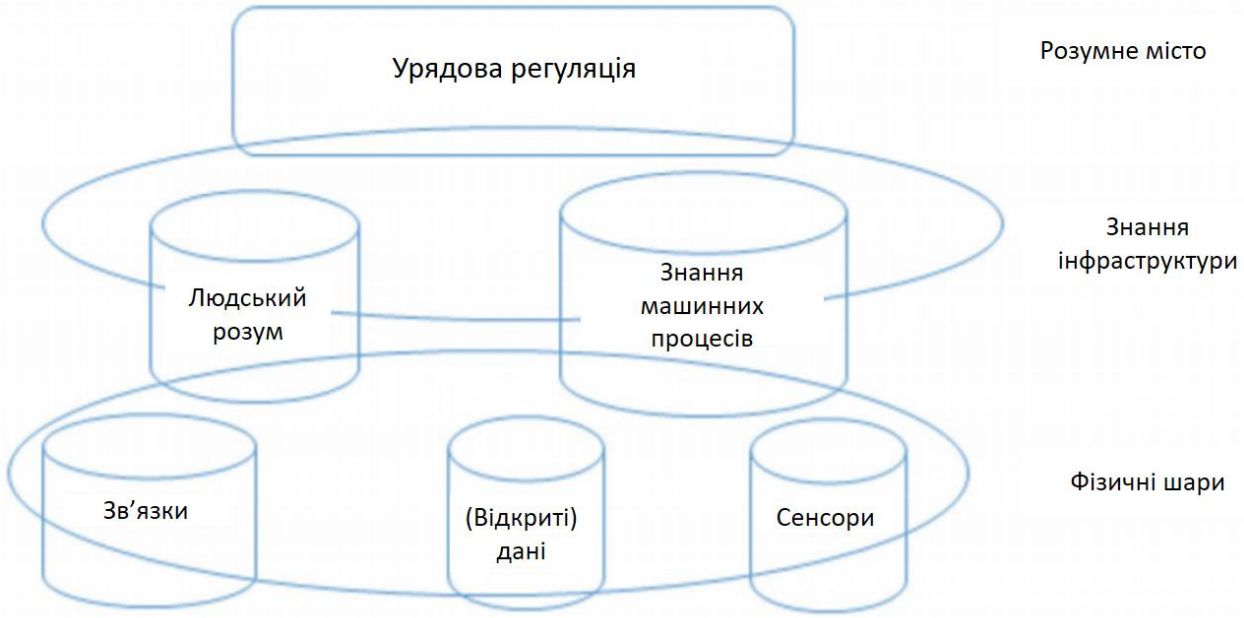


Рисунок 1.3 – Інфраструктура знань ролі в розумному місті прийняті за Лауріні

Отже, відкриті дані від сенсорів та інших з'єднань, що стосуються ініціатив розумних міст, використовуються як інформація для надання знань розробникам інформаційної інфраструктури розумного міста у створенні рішення про те, як зробити міста «розумнішими».

1.4 Системи рекомендацій як інструмент для побудови інформаційної інфраструктури розумного міста

Системи рекомендацій з'явилися з області відкриття знань і використовувались для виявлення закономірностей в великих масивах даних. У системах рекомендації застосовується фільтрування даних у поданні інформації

користувачам. Вони характеризують тип програмних систем, які дають значущі рекомендації для користувачів та допомагають користувачам у прийнятті рішень. У побудові систем рекомендації використовують методи аналізу даних у наданні пропозицій на основі сукупних даних. Система рекомендації на вході отримує та агрегує вхідні дані, а на виході надає користувачеві релевантну рекомендацію.

Рекомендовані системи були розгорнуті у багатьох сферах такі як пошукові інформації, електронна комерція, когнітивна наука, веб-ресурси та багато інших. У розумних містах генерується безліч даних, які формують величезну кількість інформації. Отже, є потреба для підходу, який має можливість фільтрувати дані для покращення пошуку інформації. Відповідно, система рекомендації може бути застосована як підхід для отримання користувачами контекстної інформації, необхідної для прийняття рішення.

Існують наступні типи систем рекомендацій, що застосовуються в тому числі для побудови інформаційної платформи розумного міста (*див. Рис.1.4.*)

Рисунок 1.4 – Типи рекомендаційних систем

Існують наступні типи систем рекомендацій, що застосовуються в тому числі для побудови інформаційної платформи розумного міста:

1. Рекомендатори на основі колабораційної фільтрації (Collaborative filtering recommenders).
2. Рекомендатори на основі вмісту (Conten-based filtering).
3. Рекомендатори, що базуються на знаннях.
4. Гібридні рекомендатори.

Детальніше кожен із типів систем рекомендації буде розглянуто у *Розділі 2.* У цьому ж розділі розглянемо та дамо відповідь на інше питання: які попередні дослідження розробляли системи рекомендацій у сфері побудов розумного міста? Таким чином, зробимо огляд відповідних досліджень, які мали на меті розробити системи рекомендацій для побудови розумного міста.

Серед цих досліджень Khan та ін. (2020) запровадив систему рекомендацій для побудови розумного ринку, заснованому на мобільній хмарності та методу контекстної інформації, що надає доступні послуги громадянам. Запропонована система рекомендацій включає рівень хмарного інтерфейсу, аналітики даних, зондування контексту та рекомендації юридичних осіб (потенційні клієнти та цифрові ринки).

Крім того, Faieq та ін. (2019) запропонував систему рекомендацій з урахуванням контексту для надання складу послуг в розумних середовищах. Дослідники мали на меті впорядкувати величезну кількість доступних послуги та покращити міцну співпрацю між зацікавленими сторонами та надання відповідних послуг користувачам.

Хоаджлі та Резег (2019) розробили систему рекомендацій для розумного міста. Автори націлені на покращення цифрового рівня міста та надання послуг громадян, виходячи із контексту, підтримувані хмарним підходом та розгорнуті для підвищення масштабованості системи. Експеримент був розгорнуто для

перевірки масштабованості та інтерактивності шляхом вимірювання даних, що обмінюються, та часу відгуку.

Ще одне дослідження було проведено Хабібзаде та співавторами (2018), де аналітика даних, машинний інтелект та м'яке зондування були інтегровані для розробки рекомендованої системи в інтелектуальному режимі міст. Застосування методів полегшує отримання даних шляхом отримання лише відповідної інформації для підтримки м'якого зондування в програмах розумного міста.

Так само, Ді Мартіно та Россі (2016) запропонували архітектуру для інструменту, що рекомендує вибір транспорту у розумних містах. Інструмент "Рекомендатор" розгорнув мультимодальність та на основі транспортних засобів допомагає користувачеві у плануванні поїздки, а також пропонує пропозиції щодо паркування для всього громадського транспорту. Архітектура включає мобільний користувацький додаток, рейтинговий список маршрутів, систему рекомендацій та джерела даних (динамічний і статичний).

Яварі та ін. (2016) розробили спеціальну систему рекомендацій щодо паркування для надання послуг доставки в розумних містах. Вони використовували технологію інтернету речей (IoT – Internet of Things) для пошуку даних від датчиків надання персоналізованої інформації користувачам. Автори застосували свій підхід до розумного місця для паркування, а також провели експериментальну перевірку, щоб показати переваги їх рішення.

Абу-Ісса та ін. (2017) реалізували активний, багатоступінчастий та контекстно-орієнтований додаток-рекомендатор у розумному місті. Система рекомендацій пропонує кілька типів сервісів і активно просуває явні пропозиції щодо запитів до користувачів.

Benfares та ін. (2017) застосовали семантичну служби надання рекомендацій у розумних містах для вирішення питання щодо створення та вибору індивідуальних послуг для підтримки прийняття рішень користувачами в режимі

реального часу. Крім того, Benfares та ін. (2016) розробили персоналізовану архітектуру послуг з рекомендацій щодо туризму в розумному місті. Архітектура використовує спільне мерехтіння як спосіб надання персоналізованих туристичних рекомендаційних послуг на основі профілю користувача.

Казино та ін. (2017) розробили контекстно-орієнтований мобільний рекомендатор для оптимізації маршрутів в розумному місті. Запропонований спосіб забезпечує адаптивну рекомендацію вправ користувача на основі їх медичного стану та даних розумного міста в режимі реального часу.

Кортес-Седіель та ін. (2017) розгорнули систему рекомендацій для просування електронного врядування в розумних містах. Автори спрямовали свою увагу на усунення відсутності персоналізації послуг для конкретних користувачів та зацікавлених сторін, що ідентифікується як одна з проблем, з якою стикаються розумні міста.

Казино та ін. (2015) запропонували контекстну систему рекомендацій для розумного здоров'я. Метою дослідників є надання медичних послуг для мешканців розумного міста для підвищення їх якості життя. Схема включає в себе зондування інфраструктури для забезпечення громадян пропозиціями щодо своїх уподобань та стану здоров’я.

Негре та Розенталь-Сабру (2014) рекомендаційний підхід, заснований на вимірах розумного міста для покращення розумності міста. Їхній підхід також допомагає прогнозувати рейтинги користувачів щодо елементів розумних міст.

Нарешті, Luberg та ін. (2011) розробив додаток, що базується на правилах в розумному місті для надання спеціальних рекомендацій з використанням імовірнісних міркувань.

Наслідки розглянутих 14 досліджень свідчать про те, що, хоча оглянуті дослідження розробили рекомендацію системи для вирішення різних питань у

розумних містах, існує мало досліджень, в яких застосовувався би типовий підхід до побудови системи рекомендацій одним із 4 методів, розглянутим вище.

1.5. Висновки до розділу

В даному розділі було досліджено поняття розумного міста, дослідження наукових експериментів у сфері розробки інформаційної інфраструктури для побудови розумного міста. Було проведено аналіз та порівняння аналіз та порівняння існуючих систем та наукових досліджень в даній предметній області. Також було розглянуто поняття рекомендаційної системи як засіб побудови персоналізованих пропозицій для жителів розумного міста для допомоги організації їх соціально-культурного життя.

РОЗДІЛ 2. МЕТОДИ ПОБУДОВИ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ ДЛЯ РОЗРОБКИ ІНФОРМАЦІЙНОЇ ІНФРАСТРУКТУРИ РОЗУМНОГО МІСТА

2.1. Огляд поняття систем рекомендації та приклади їх застосування

Вибухове зростання кількості доступної цифрової інформації та кількості відвідувачів Інтернету створили потенційний виклик перевантаження інформацією, що заважає своєчасному доступу до предметів, що цікавлять в Інтернеті. Інформація у пошукових системи, такі як Google, DevilFinder та Altavista частково вирішили цю проблему, але пріоритети та персоналізація (де система відображає наявний вміст за інтересами та уподобаннями користувача) інформації відсутні. Це має збільшив попит на системи рекомендації як ніколи раніше.

Системи рекомендації – це інструменти для взаємодії з великими і складними джерелами інформації. Вони забезпечують персоналізоване представлення різних об’єктів та надають пріоритет тим, що можуть зацікавити користувача. Рекомендаційні системи також визначають як стратегії стратегія прийняття рішень для користувачів у складних інформаційних середовищах [6].

Крім того, системи рекомендації були визначені з точки зору електронної комерції як інструмент, який допомагає користувачам шукати в записах знань, пов’язаних із інтересами та уподобаннями користувачів [7]. Системи рекомендацій були визначені як засіб надання допомоги та посилення соціального процесу використання рекомендацій інших користувачів для прийняття рішення, коли немає достатньо особистого знання або досвіду альтернативних варіантів [8].

Ціллю найбільш розвинених систем рекомендації є створення персоналізованої рекомендаційної системи, яка буде враховувати як особисті вподобання користувача на основі попередніх дій користувача, так і пропонувати

такі, що можуть його потенційно зацікавити на основі технологій, що враховують подібність поведінки користувача з іншими.

Успіх будь-якої системи рекомендацій значною мірою залежить від її здатності представляти поточні інтереси користувача. Точні моделі необхідні для отримання відповідних і точних рекомендацій за допомогою будь-яких методів прогнозування.

Дослідження рекомендаційних систем охоплюють сценарії в багатьох середовищах з доступом до інформації, в якому як користувачі, так і власники інформаційних ресурсів можуть отримати вигоду від презентації персоналізованих варіантів. Дана сфера набула великого інтересу за останнє десятиліття, частково спричинена премією Netflix (Bennett & Lanning, 2007р.), про що свідчить швидке зростання кількості наукових конференцій, присвячених дослідженню методів побудови рекомендаційних систем.

Персоналізовані рекомендації є важливою частиною багатьох онлайн-сервісів електронної комерції, таких як Amazon.com, Netflix, Pandora, тощо. Такий вдалий досвід застосування дає можливість для розширення сфери застосування рекомендаційних систем в нові та складні сфери.

Рекомендовані системи вигідні як постачальникам послуг, так і користувачам. Вони зменшують трансакційні витрати на пошук і відбір предметів в середовищі інтернет-покупок. Також доведено, що системи рекомендацій вдосконалюють процес та якість прийняття рішень. У налаштуваннях електронної комерції рекомендаційні системи збільшують доходи через те, що вони є ефективними засобами продажу великої кількості товарів.

В наукових бібліотеках, рекомендуючі системи допомагають користувачам, дозволяючи їм перейти за межі пошуку за каталогами. Саме тому важливо не обходити стороною факт необхідності використовувати ефективні та точні методи

рекомендацій в рамках системи, які забезпечить релевантність та надійність рекомендації для користувачів не можна надмірно наголошувати.

Становлення рекомендаційних систем як напрямку набуло популярності останні 14 років. У настійній статті Ресника та Варіана, автори описують систему рекомендацій таким чином: «У типовій системі рекомендацій люди надають рекомендації як вхідні дані, які потім система агрегує та спрямовує на відповідних одержувачів. У деяких випадках первинна трансформація відбувається в агрегації; в інших цінність системи полягає в її здатності зробити хороші матчі між рекомендаціями та тими, хто шукає рекомендації ». [9]

Зауважимо, що це визначення робить акцент на системі рекомендацій як підтримці співпраці між користувачами. Пізніше дослідники розширили своє визначення включення систем, які пропонують цікаві предмети, незалежно від того, як виробляються ці рекомендації: “Будь-яка система, яка виробляє індивідуалізовану рекомендації як вихідні або мають ефект керівництва користувач персоналізовано під цікаві або корисні об’єкти у великому просторі можливих варіантів» [9].

Більш формально проблема рекомендацій можна сформулювати так: Нехай C – множина всіх користувачів, і нехай S є набором усіх можливих елементів, які можна рекомендувати. Нехай u - функція, яка вимірює корисність елемента s до користувача c, тобто u: CxS⇒R, де R є а повністю впорядкований набір (наприклад, неотримані цілі числа) або реальні числа в певному діапазоні). Тоді, для кожного користувача c∈C, ми хочемо вибрати такий елемент s`∈S, що забезпечує максимальну корисність користувача.

Це визначення відкриває поле рекомендації системи для будь-якого додатка, що обчислює корисність конкретного користувача, що охоплює багато проблем, про які зазвичай думають як про додатки для пошуку в базі даних або інформації. Навіть це широке визначення може бути занадто вузьким, як деякі рекомендатори

можуть працювати на конфігураціях - на противагу до фіксованого набору S усіх предметів - і інші роблять рекомендації для груп.

Визначення також може бути трохи оманливим - у багатьох рекомендаційних системи не розраховують явно утиліти, коли вони складають класифікований список рекомендованих елементів.

Автори обережно заявляють, що мета – вибір елемента з найкращою утилітою, не обов'язково для обчислення утиліти явним чином.

З цих міркувань випливають два основні принципи. Виділяють, що розрізняють дослідження рекомендаційних систем:

- персоналізовані рекомендації. Рекомендації, які вона виробляє, призначені оптимізувати досвід роботи одного користувача, не для цього представляють груповий консенсус для всіх;
- рекомендаційна система призначена для допомоги користувачу обрати серед дискретних варіантів. Усі елементи вже заздалегідь відомі і не породжується на замовлення.

Персоналізаційний аспект систем рекомендування цього напрямку досліджень найбільш сильно відрізняється від того, що прийнято розуміти як дослідження в пошукових системах і інших програмах пошуку інформації. У пошуковій системі або іншій системах пошуку інформації, ми очікуємо набір результатів, що відповідають певному запиту, будуть однаковими незалежно від того, хто його видав.

Багато систем рекомендацій добиваються персоналізації, підтримуючи профілі користувачів (довгострокові або короткострокові) або за заявленими уподобаннями. Інші досягають персоналізованого результату за допомогою розмовної взаємодії.

2.2. Етапи побудови систем рекомендації

Етап 1. Збір інформації. На цьому етапі збирається відповідна інформація про користувачів для створення користувацького профілю або моделі для завдань передбачення, включаючи атрибути користувача, поведінку або вміст ресурсів, до яких користувач отримує доступ. Рекомендаційний агент не може функціонувати точно, доки Профіль користувача / модель не була добре побудована. Система потребує знати якомога більше про користувача, щоб надати релевантну рекомендацію.

При створенні систем рекомендації використовуються різні типи отримання інформації про користувача – як найбільш зручний та високоякісний явний відгук, який включає явні вказівки користувачів щодо їх інтересу до товару, так і неявний зворотний зв'язок шляхом опосередкованого висновку про переваги користувачів через спостереження за поведінкою користувача [10].

Також може використовуватись гібридний зворотній зв'язок, який можна також отримати за допомогою комбінації використання обох явних та неявний зворотних зв'язків.

Наприклад, на платформі електронного навчання профіль користувача – це сукупність особистої інформації, пов’язаної з конкретним користувачем. Ця інформація включає когнітивні навички, інтелектуальні здібності, стилі навчання, інтерес, уподобання та взаємодія з системою.

Профіль користувача зазвичай використовується для отримання необхідної інформації для побудови моделі користувача. Таким чином, профіль користувача описує просту модель користувача.

Схематичне зображення етапів побудови систем рекомендації можна розглянути на *Рис.2.1.*

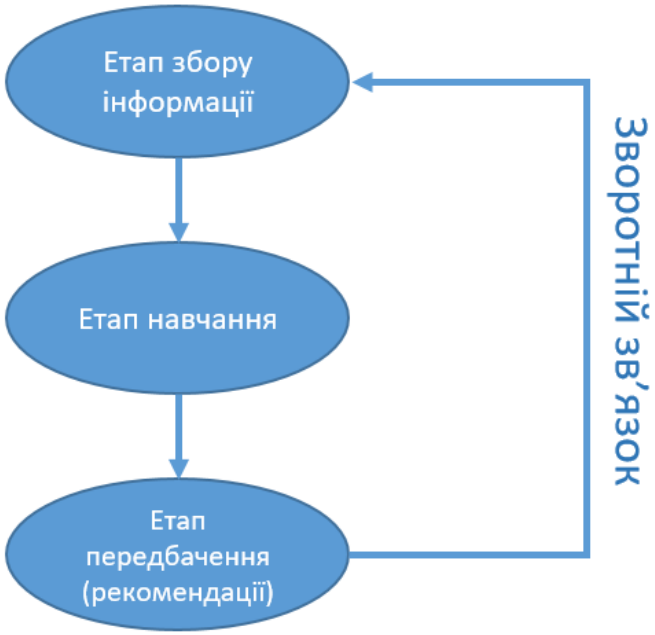


Рисунок 2.1. – Етапи побудови системи рекомендації

Явний зворотній зв'язок. Зазвичай система запрошує користувача через систему інтерфейс для надання оцінок для елементів для побудови і вдосконалити свою модель. Точність рекомендацій залежить від кількості оцінок, наданих користувачем. Єдиним недоліком цього методу є, він вимагає зусиль від користувача, а також користувачі не завжди готові забезпечити достатньо інформації.

Незважаючи на те, що явний зворотний зв'язок вимагає більше зусиль від користувача, його все все ще розглядається як спосіб забезпечення отримання більш надійних даних, оскільки це не передбачає вилучення інформації та закономірностей з дій, що виконуються користувачем, а також забезпечує прозорість процесу рекомендацій, що призводить до дещо вищої якості сприйняття рекомендацій та більшої довіри до рекомендації [11].

Неявний зворотній зв'язок. Система автоматично визначає уподобання користувача, відстежуючи різні дії користувачів, такі як історія покупок, історія

навігації та час перебування на деяких веб-сторінках, посилання, за якими слідує користувач, вміст електронної пошти та натискання кнопок серед інших.

Неявний зворотній зв'язок зменшує навантаження на користувачів, роблячи висновки про переваги користувачів із їх поведінки в системі. Метод не вимагає зусиль користувачеві, проте такий спосіб отримання інформації про користувача є менш точним. З іншого боку, було аргументовано, що неявні дані переваг насправді можуть бути більш об'єктивними, оскільки не існує упередженості, що виникає в результаті реагування користувачів на соціально бажаним способом і не виникає проблем із самовідчуттям або будь-яка потреба в підтримці іміджу для інших [12].

Гібридний зворотний зв'язок. Сильні сторони як неявного, так і явного зворотного зв'язку можуть бути об'єднані в гібридну систему, щоб мінімізувати їх слабкі сторони та отримати найкращу систему. Цього можна досягти використовуючи неявні дані як перевірку явного рейтингу або дозволяючи користувачеві давати явний зворотний зв'язок лише тоді, коли він вирішив виявити явний інтерес.

Етап 2. Навчання. На цьому етапі застосовуються алгоритм навчання для фільтрації та використання користувацького особливості зворотного зв'язку, зібраного при етапі збору інформації.

Етап 3. Етап прогнозування / рекомендації. На даному етапі здійснюється рекомендація або передбачення тих елементів, в яких користувач може бути зацікавлений. Це можна зробити безпосередньо на основі набору даних**,** зібраних на етапі збору інформації, який може бути заснований на пам'яті чи моделі, або через спостереження за поведінкою користувача. На *Рис. 2.1.* схематично проілюстровані описані вище етапи побудови та використання систем рекомендацій.

2.3. Типи рекомендаційних систем

Для побудови рекомендаційних систем використовують різні підходи та принципи отримання результату рекомендації (*див.Рис.2.2*). Відповідно до цих підходів системи рекомендації поділяють на наступні типи: рекомендатори на основі вмісту (Content-based recommenders), рекомендатори на основі колаборацийної фільтрації (Collaborative fltering recommenders) та гібридні рекомендатори.

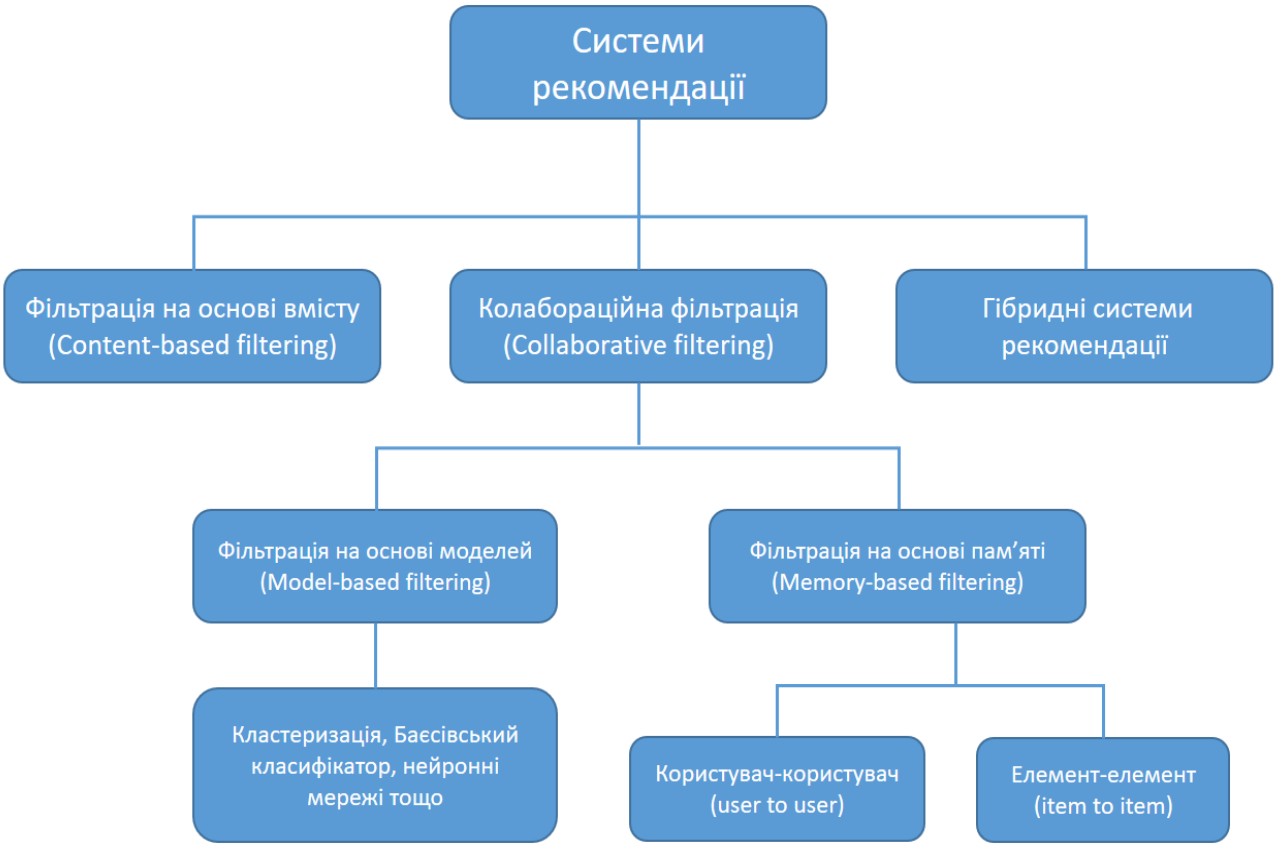


Рисунок 2.2 – Підходи до побудови систем рекомендацій

Використання ефективних методів рекомендацій є дуже важливою складовою для побудови якісної системи рекомендації, яка має на меті надавати корисні та релевантні рекомендації своїм окремим користувачам. Це пояснює важливість розуміння особливостей та потенціалів різних рекомендаційних систем.

На Рис. 2.2 зображено різні підходи, які використовуються для побудови систем рекомендацій.

Для більш детального розгляду методів побудови рекомендаційних систем спочатку формалізуємо задачу. У нас є множина користувачів $u \in U$, множина об’єктів $i \in I$ (об’єкти, що будуть рекомендуватись) та множина подій $(rui, u, i, ...) \in \mathcal{D}$ (дії, які користувачі здійснюють з об’єктами). Кожна подія задається користувачем u, об’єктом i, його результатом rui і, можливо, ще якими-то характеристиками.

Нам потрібно:

– Передбачити вподобання користувача:

$$\hat{rui} = Predict\ (u, i, ...) \approx rui$$

– Здійснити персональну рекомендацію:

$$u \longmapsto (i1, ..., iK) = Recommend\ K\ (u, ...)$$

– Визначити подібні об'єкти:

$$u \longmapsto (i1, ..., iM) = \text{Similar}\ M\ (i)$$

2.3.1. Фільтрація на основі вмісту (Content-based filtering)

Фільтрація на основі вмісту – це підхід побудови рекомендаційних систем, який будується на основі контенту, що розміщений в системах та додатках. При застосуванні цього методу застосовується аналіз ознак об’єктів, що можуть рекомендуватися користувачам.

Якщо середовища, в яких застосовуються рекомендації такі як, наприклад, веб-сторінки, портали для публікацій та новин, фільтрація на основі вмісту є найбільш успішною.

В системах рекомендації, які побудовані за допомогою фільтрації на основі вмісту, рекомендації здійснюються на основі профілів користувача з використанням ознак, вилучених із характеристик елементів, які користувач оцінював у минулому [13,14].

Ці методи дають рекомендації вивчення базової моделі за допомогою будь-якого статистичного аналізу або техніки машинного навчання. Техніка фільтрації на основі вмісту не потребує профілю інших користувачів, оскільки вони не впливають на рекомендації.

Крім того, якщо профіль користувача змінився, система рекомендації на основі вмісту все ще має можливість коригувати свої рекомендації протягом дуже короткого періоду часу. Недоліком цієї техніки є необхідність мати поглиблені знання та опис особливостей об'єктів, що можуть рекомендуватися користувачеві.

Рекомендатори на основі вмісту розгортають інший підхід, де рекомендація елементів не узгоджується з оцінками попередніх користувачів. Ця подібність вимірюється на основі характеристики товару . Таким чином, рекомендований на основі вмісту заснований на уявленні, що інформація елемента можна легко визначити за категоріальними типами даних.

Цей підхід передбачає значні знання в області, які можуть бути складним у підтримці. У рекомендації на основі вмісту деякі типи інформації, такі як мультимедіа непросто аналізувати. Основне обмеження цієї системи полягає в тому, що вона не рекомендує користувачеві інформацію, за винятком випадків, коли користувачеві була цікава подібна інформація в минулому [15].

Альтернативою рекомендацій на основі вмісту є впровадження підходу на основі вмісту, де програми можуть бути рекомендовані для встановлення, якщо потрібні їх пристрої (передбачається, що ця інформація надається для кожного додатка) є «сумісними» з місцевими конфігурація шлюзу.

Застосовуючи підхід на основі фільтрації на основі вмісту, рекомендовані елементи визначаються на основі подібності між інформацією про профіль локального шлюзу (наприклад, у терміни встановлених пристроїв) та інформацію про профіль програм, доступних, наприклад, на ринку у хмарі.

Фільтрація на основі вмісту є одним із найпоширеніших методів побудови систем рекомендацій. При дослідженні різних наукових статей та матеріалів можна помітити, що в науковому середовищі існує два підходи побудови рекомендаційних систем на основі вмісту.

Тому розглянемо кожен із них і порівняємо їх переваги та недоліки. Таким чином, можна виділити два підходи, що застосовуються при побудови систем рекомендацій на основі вмісту:

- **Підхід 1** – Аналіз опису лише характеристик вмісту/контенту;
- **Підхід 2** – Створення профілю користувача та профілю елементів із вмісту, оціненого користувачем.

Підхід 1 – Аналіз опису лише характеристик вмісту/контенту. При застосуванні такого підходу при побудові системи рекомендації аналізують лише характеристики елементів, що можуть бути рекомендовані користувачеві. Таким

чином, система рекомендує користувачеві елементи, що схожі на ті елементи, які сподобалися користувачеві раніше.

На етапі побудови моделі система спочатку виявляє схожість між усіма парами елементів (*див. Рис. 2.3*), потім використовує найбільш подібні елементи до вже оцінених елементів користувача, щоб сформувати список рекомендацій на етапі рекомендацій.

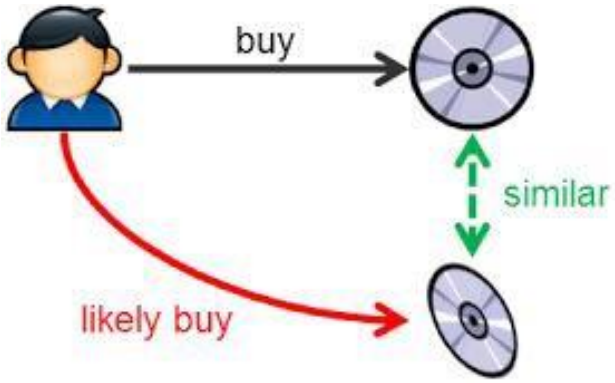


Рисунок 2.3 – Ілюстрація підходу рекомендації на основі вмісту, при якому враховуються лише характеристики елементів

Наприклад, розглянемо систему рекомендації, у якій об’єктами рекомендації є фільми. Якщо хтось дивиться Edge of Tomorrow, система може рекомендувати Looper на основі подібності. Для того, щоб пропонувати релевантні рекомендації, система рекомендації, побудована на основі вмісту використовує різні методи, щоб знайти «схожість» між елементами. Він може використовувати як векторно-просторові моделі, такі як TF-IDF (TF — term frequency, IDF — inverse document frequency), так і імовірнісні моделі, такі як Наївний баєсівський класифікатор, дерева рішень [15] або нейронні мережі для моделювання взаємозв'язку між різними елементами.

Одним із поширених методів визначення подібності між елементами, що можуть рекомендуватись, є метод TF-IDF. При використанні цього методу кожен елемент буде представлений вектором TF-IDF.

Метод TF-IDF найчастіше використовується в області розпізнавання та обробки природньої мови (NLP). Він використовується для пошуку інформації з метою вилучення ознак. Таким чином, він вираховує частоту кожного слова в документі і зважуєте важливість кожного слова, обчислює оцінку для цього документа.

TF (term frequency) розраховується як частота слова в поточному документі до загальної кількості слів у документі. Таким чином, за допомогою цього кроку визначається частоту слова в тому чи іншому документі і надається більша вага, коли частота більша, для нормалізації ділиться на довжину документа:

$$Tf(t) = \frac{\text{Частота появи слова } t \text{ в документі}}{\text{Загальні кільскість слів у документі}}$$

IDF (inverse document frequency) розраховуєтсья як загальна кількість документів до частоти зустрічальності документів, що містять слово. Це визначає рідкість слова, таким чином слово, що зустрічається в документі менше, збільшує IDF. Це допомагає дати більший бал рідкісним термінам у документах:

$$Idf(t) = log_{10} \frac{\text{Загальне число документів}}{\text{Кількість документів, що містять слово } t}$$

Таким чином, TF-IDF – це міра, що використовується для оцінки того, наскільки важливим є слово для документа в системі документів. Важливість слова

зростає пропорційно кількості випадків, коли слово з'являється в документі, але компенсується частотою слова в ньому.

Оскільки метод в значній мірі покладається на опис для розрізнення кожного товару, опис повинен глибше відображати деталі товару, тобто назву, резюме, слогани, жанр так, щоб він надавав якомога більше інформації про елемент, що може рекомендуватися.

Після визначення значення TF-IDF для тегів, ми можемо створити TF-IDF вектори – вектори ключових слів для кожного елемента.

Наступним кроком у побудові рекомендації на основі вмісту є обчислення схожості векторів елементів. Для обчислення цієї схожості використовується формула:

$$similarity(user, item) = \frac{features(user) \cap features(item)}{features(user) \cup features(item)}$$

Подібно до спільної фільтрації, існують різні типи метрик подібності, наприклад:

1. **Евклідова відстань.**

Евклідова відстань між двома точками - це довжина відрізків, що їх з'єднують. Наш евклідовий простір у цьому конкретному випадку – це позитивна частина площини, де осі є ранжированими елементами, а точки представляють оцінки, які конкретна людина дає обом предметам.

Двоє людей належать до певного простору переваг тоді і лише тоді, коли вони класифікували два елементи, що визначають простір переваг. Отже, ми визначаємо простір переваг для кожної пари різних елементів, а точки в цьому просторі переваг задаються людьми, які класифікували два елементи. Вона розраховується за формулою:

$$Euclidean\ Distance = \sqrt{(x_1 - y_1)^2 + \cdots\ (x_N - y_N)^2}$$

2. Подібність за косинусом (визначає лінійну залежність векторів).

При використанні метрики подібності за косинусом алгоритм знаходить косинус кута між вектором профілю та вектором елемента *(Див. Рис2.4.)*

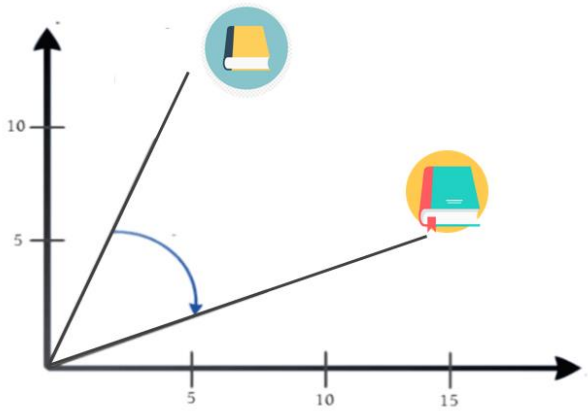


Рисунок 2.4 – Ілюстрація метрики подібності за косинусом

Виходячи із значення косинуса, яке коливається від -1 до 1, елементи розташовані в порядку зменшення, а для рекомендацій використовується один із двох наведених нижче підходів:

– топ-n підхід: де рекомендовано перші n елементів;
– підхід за шкалою рейтингу: де рекомендуються всі елементи, які перевищують встановлений поріг.

Подібність за косинусом розраховується за формулою:

$$\cos(\theta) = \frac{A\ B}{\|A\|\ \|B\|} = \frac{(\sum_{i=1}^{n} A_i B_i)}{\sqrt[2]{\sum_{i=1}^{n}(A_i)^2}\ \ \sqrt[2]{\sum_{i=1}^{n}(B_i)^2}}$$

1) Кореляція Пірсона

Кореляція Пірсона показує, наскільки два елементи співвідносяться або подібні, це співвідношення можна розрахувати за наступною формулою:

$$sim(u,v) = \frac{\sum(r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum(r_{ui} - \bar{r}_u)^2}\ \ \sqrt{\sum(r_{vi} - \bar{r}_v)^2}}$$

Основним недоліком цього алгоритму є те, що він обмежується рекомендуванням предметів одного типу.

Підхід 2: Створення профілю користувача та профілю елемента з вмісту, оціненого користувачем (*див. Рис. 2.5*). Цей підхід використовує опис або атрибути елементів, з якими користувач взаємодіяв, щоб рекомендувати подібні елементи. Це залежить лише від попереднього вибору користувача, що робить цей метод надійним, щоб уникнути проблеми холодного запуску.

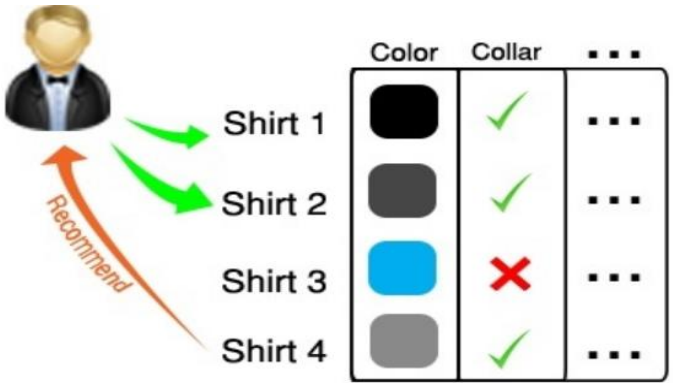


Рисунок 2.5 – Ілюстрація підходу рекомендації на основі вмісту, при якому створюється профіль користувача

Припустимо, що користувач дивиться певний жанр, йому будуть рекомендовані фільми, що відповідають цьому конкретному жанру (*див. Рис. 2.6*). Назва, рік випуску, режисер, акторський склад також допомагають визначити подібний зміст фільму.

Рисунок 2.6 – Взаємодія користувача із системою при підході на основі формування профілю

У цьому підході вміст продукту вже оцінюється на основі уподобань користувача (Профілю користувача), тоді як жанр елемента – це неявні особливості, за якими він буде використовуватися для побудови профілю товару (*див. Рис. 2.7*).

Потім оцінюється оцінка товару за допомогою обох профілів, і можна зробити рекомендацію. Подібно до підходу 1, у цьому підході також буде використана методика TF-IDF.

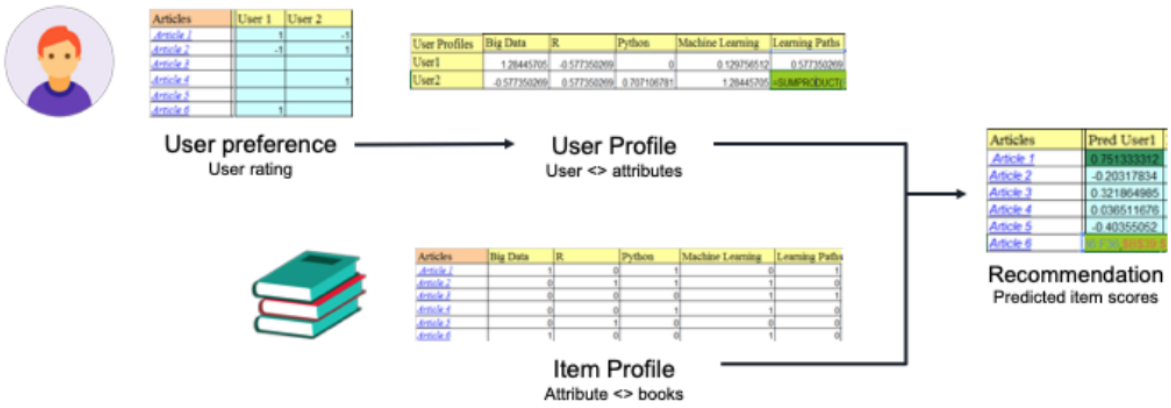


Рисунок 2.7 – Схема формування профілю користувача та взаємодії з користувачем в системі рекомендації

Для того, щоб сформувати профіль користувача, зазвичай формують таблицю рейтингів (рейтинг користувачів), профіль предметів (характеристики елементів, наприклад жанри, автори і рік випуску книг) – це єдиний матеріал, який ми маємо:

– рейтингова таблиця: взаємозв’язок між користувачем та книгою;
– профіль елемента: відносини між атрибутом і книгою;
– профіль користувача: взаємозв'язок "користувач-атрибут".

Таким чином створюється профіль користувача, щоб ми могли зрозуміти, яким елементам користувачі насправді віддають перевагу

Таким чином, за допомогою профілю користувача ми можемо передбачити всі оцінки предметів конкретного користувача на основі його / її профілю користувача та профілю товару.

Переваги та недоліки обох підходів розглянуто в Таблиці 2.1.

Таблиця 2.1 – Переваги та недоліки різних підходів

	Переваги	Недоліки
Підхід 1	– На відміну від спільної фільтрації, якщо елементи мають достатньо описів, ми уникаємо “проблеми з новими елементами”. – Представлення вмісту різноманітні, і вони відкривають можливості для використання різних підходів, таких як: методи обробки тексту, використання семантичної інформації, умовиводів тощо ... – Зробити більш прозору систему легко: ми використовуємо той самий вміст, щоб пояснити рекомендації.	– Системи рекомендації, засновані на даному підході, мають тенденцію до надмірної спеціалізації: вони рекомендуватимуть предмети, подібні до тих, що вже спожиті, з тенденцією до створення "бульбашки фільтра".
Підхід 2	– Незалежність від користувача: для спільної фільтрації потрібен рейтинг інших користувачів, щоб знайти схожість між користувачами, а	– Обмежений аналіз вмісту: якщо вміст не містить достатньо інформації для точної дискримінації

Продовження таблиці 2.1.

	потім дати пропозицію. Натомість за методом, що базується на вмісті, потрібно лише проаналізувати елементи та профіль користувача для отримання рекомендацій. – Прозорість: метод спільної роботи дає вам рекомендацію, оскільки деякі невідомі користувачі мають такий самий смак, як ви, але метод, заснований на вмісті, може сказати вам, що вони рекомендують вам елементи на основі яких функцій. – Ніякого холодного початку: на противагу спільній фільтрації, нові елементи можна пропонувати, перш ніж оцінювати значна кількість користувачів.	предметів, рекомендація не буде точно в кінці. – Надмірна спеціалізація: метод, що базується на вмісті, забезпечує граничний ступінь новизни, оскільки він повинен відповідати особливостям профілю та предметів. Повністю досконала фільтрація на основі вмісту може не припустити нічого «здивованого». – Новий користувач: коли для створення надійного профілю користувача недостатньо інформації, рекомендація не може бути надана правильно.

2.3.2. Колаборaційна фільтрація (Collaborative fltering)

На відміну від систем рекомендації на основі вмісту, системи рекомендації, що побудовані за допомогою колаборативної фільтрації використовують взаємодію користувачів та перевагу інших користувачів для фільтрування предметів, що можуть зацікавити.

Це один з найбільш часто використовуваних алгоритмів у галузі, оскільки він не залежить від будь-якої додаткової інформації. Існують різні типи спільних методів фільтрації, і ми детально розглянемо їх нижче.

Рекомендатор на основі колаборaційної фільтрації – найпопулярніший підхід, завдяки якому рекомендації виконуються і в реальному житті , оскільки за його основу взято те, що користувач A рекомендує інформацію користувачеві B. Це трактується в системах як ситуація, коли користувачі частіше цікавляться інформацією, яка вже сподобалась іншим користувачам з подібним інтересом. Рекомендатор, що співпрацює, допомагає обмінюватися знаннями та / або досвідом серед користувачів, які мають подібний інтерес. Недолік цього методу полягає в тому, що нова інформація не рекомендується поки він не матиме достатньо оцінок користувачів.

Базовим інструментом при підході колаборативної фільтрації є матрична факторизація (*див. Рис.2.8*). Мета полягає в тому, щоб заповнити невідоме в матриці взаємодій між елементами користувача (назвемо це RR).

Нехай у нас є дві матриці UU і II, такі, що U \ раз IU × I дорівнює RR у відомих записах. Використовуючи продукт U \ раз IU × I, ми також матимемо значення для невідомих записів RR, які потім можна використовувати для формування рекомендацій.

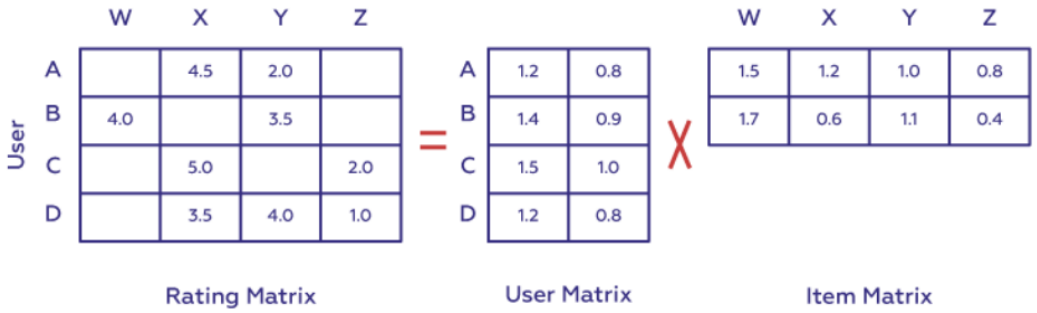


Рисунок 2.8 – Матрична факторизація

Розумний спосіб знайти матриці UU та II – це використання алгоритмів машинного навчання, таких як дерева прийняття рішень чи нейронної мережі. Цікавий спосіб розгляду цього методу полягає в тому, щоб мислити про нього як про узагальнення класифікації та регресії.

Хоча вони складніші та розумніші, вони повинні мати достатньо інформації для роботи, тобто холодний старт для нових веб-сайтів електронної комерції та нових користувачів.

Існує два підходи (*Див. Рис. 2.9*) для побудови систем рекомендації за допомогою колаборацийного фільтрування:

– колаборацийна фільтрація на основі пам'яті;
– колаборацийна фільтрація на основі моделей.

Другий підхід визначає кластери елементів, які були оцінені певним користувачем, і використовує їх для прогнозування взаємодії користувача з подібним елементом. Методи, засновані на пам'яті, прості у впровадженні та прозорі, але вони стикаються з великими проблемами при великих розріджених матрицях, оскільки кількість взаємодій між елементами користувача може бути занадто малою для створення високоякісних кластерів.

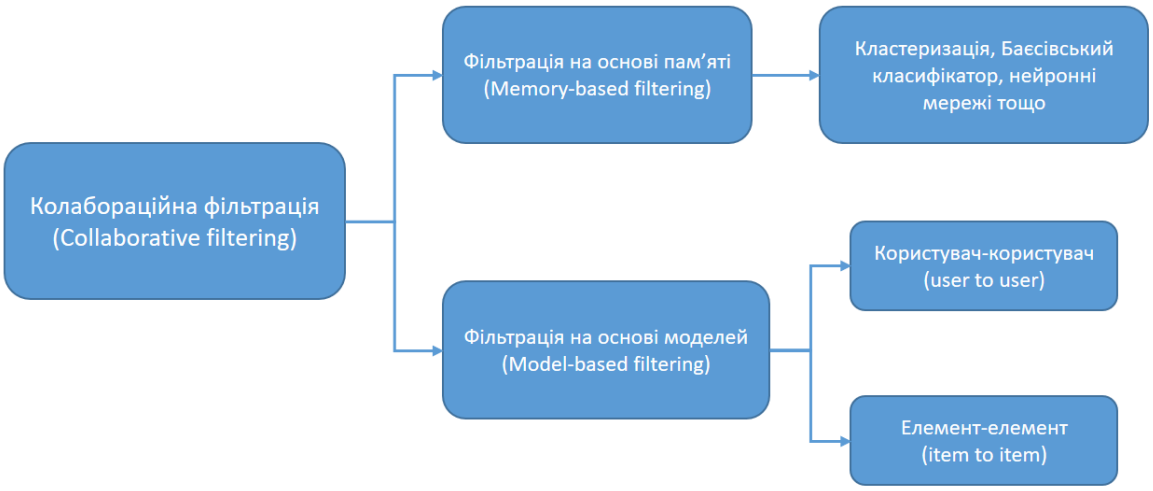


Рисунок2.9 – Типи рекомендаторів за принципом колабраційної фільтрації

Методи колаборційної фільтрації, що будуються **на основі пам'яті**, пропонують два підходи:

- «користувач – користувач» («user to user», user-based clustering) - ідентифікувати кластери користувачів;
- «елемент – елемент» («item to item», item-based clustering) - використовувати взаємодії одного конкретного користувача для прогнозування взаємодії кластера.

Колаборaційна фільтрація «користувач - користувач». Цей алгоритм спочатку знаходить показник подібності між користувачами. Виходячи з цього показника подібності, він вибирає найбільш схожих користувачів і рекомендує елементи, які сподобалися або були обрані подібними користувачами (*Див. Рис. 2.10*).

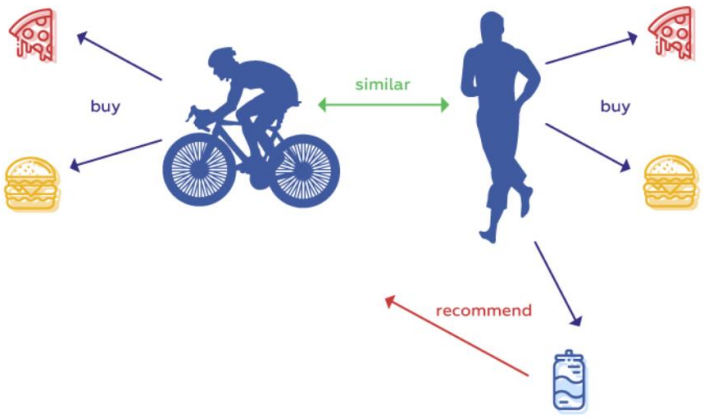


Рисунок 2.10. – Колаборативна фільтрація «користувач-користувач»

Цей алгоритм знаходить схожість між кожним користувачем на основі рейтингів, які вони раніше давали певним елементам. Передбачення елемента для користувача u обчислюється обчисленням зваженої суми оцінок користувачів, наданих іншими користувачами, на елемент i.

Тепер у нас є рейтинги для користувачів у векторному профілі, і виходячи з цього, ми маємо передбачити рейтинги для інших користувачів. Для цього виконуються наступні кроки: для прогнозів нам потрібна схожість між користувачем u та v. Ми можемо використовувати кореляцію Пірсона. Спочатку ми знаходимо елементи, оцінені як користувачами, так і на основі оцінок, обчислюється співвідношення між користувачами. Прогнози можна обчислити, використовуючи значення подібності. Цей алгоритм, перш за все, обчислює схожість між кожним користувачем, а потім на основі кожної подібності обчислює прогнози.

Користувачі, які мають більш високу кореляцію, мають тенденцію бути подібними. Цей алгоритм займає досить багато часу, оскільки передбачає обчислення схожості для кожного користувача, а потім обчислення прогнозу для кожної оцінки схожості. Одним із способів вирішення цієї проблеми є вибір лише

кількох користувачів (сусідів) замість усіх для прогнозування, тобто замість того, щоб робити прогнози для всіх значень подібності, ми вибираємо лише кілька значень подібності. Існують різні способи вибору сусідів:

1) обрання порогової схожості та обрання усіх користувачів, що перевищують це значення;
2) випадкове обрання користувачів, упорядкування сусідів у порядку зменшення їх значення подібності та вибір користувачів топ-N.

В алгоритмах, що вимірюють схожість між користувачами, передбачення елемента для користувача u обчислюється шляхом обчислення зваженої суми оцінок користувачів, наданих іншими користувачами елементу i. Прогноз Pu, i дається за формулою:

$$P_{u,i} = \frac{\sum_v (r_{i,N} * S_{u,N})}{\sum_v (S_{u,v})},$$

де $P_{u,i}$ – передбачення товару;

$R_{v,i}$ – рейтинг, наданий користувачем v елементу i.

Колаборaційна фільтрація «елемент-елемент». У цьому алгоритмі ми обчислюємо схожість між кожною парою елементів. Цей алгоритм працює аналогічно колаборaційному рекомендатору «користувач-користувач» лише з невеликою зміною - замість взяття зваженої суми оцінок "сусіди-користувачі" ми беремо зважену суму оцінок "предмет-сусід". Спочатку отримуємо схожість між парами елементів (*див. Рис. 2.11*).

Тепер, коли ми маємо схожість між кожним елементом та рейтингами, робляться прогнози та виходячи з цих прогнозів, рекомендуються подібні елементи. Давайте розберемося це на прикладі. Тут середній рейтинг елемента - це середнє

значення всіх оцінок, наданих певному товару (порівняйте його з таблицею, яку ми бачили при фільтрації користувачів-користувачів). Замість того, щоб знайти схожість між користувачем і користувачем, як ми бачили раніше, ми знаходимо подібність предмета-елемента.

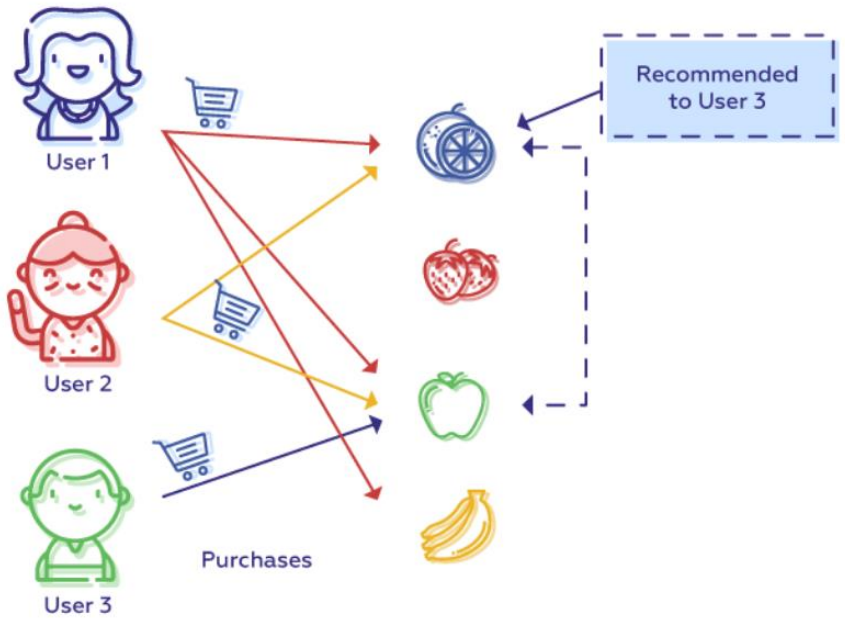


Рисунок 2.11 – Колаборативна фільтрація «елемент-елемент»

Алгоритм знаходить схожість між кожною парою предметів і, виходячи з цього, рекомендує подібні предмети, які сподобались користувачам у минулому.

Цей алгоритм працює подібно до спільної фільтрації між користувачем та користувачем, лише з невеликими змінами - замість того, щоб брати зважену суму оцінок “сусідів-користувачів”, ми беремо зважену суму оцінок “сусідів-елементів”. Прогноз дається:

$$P_{u,i} = \frac{\sum_N (S_{i,N} * R_{u,N})}{\sum_N (|S_{i,N}|)}$$

Тепер ми знайдемо подібність між предметами:

$$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\ \vec{j}}{\|\vec{i}\|_2 * \|\vec{j}\|_2}$$

Подібно до фільтрації на основі вмісту, для знаходження подібності при побудові колаборативного рекомендатора застосовують також і інші метрики оцінки подібності, такі як кореляція Пірсона та Евклідова відстань.

Особливість цього методу полягає в тому, що він повинен враховувати поняття «холодного старту» (Cold Start). Cold Start для системи означає, що в набір даних вводиться новий користувач.

Оскільки немає історії цього користувача, система не знає уподобань цього користувача. Одним із основних підходів для вирішення цієї проблеми є застосування стратегії, заснованої на популярності, тобто рекомендувати найпопулярніші продукти. Їх можна визначити за популярністю останнім часом загалом чи регіонально.

Як тільки ми дізнаємося уподобання користувача, рекомендувати продукти стане простіше. З іншого боку, продукт Cold Start означає, що новий продукт запускається на ринок або додається до системи. Дії користувача найважливіші для визначення вартості будь-якого товару.

Чим більше взаємодії отримує продукт, тим простіше нашій моделі рекомендувати цей товар потрібному користувачеві. Ми можемо використовувати фільтрацію на основі вмісту для вирішення цієї проблеми.

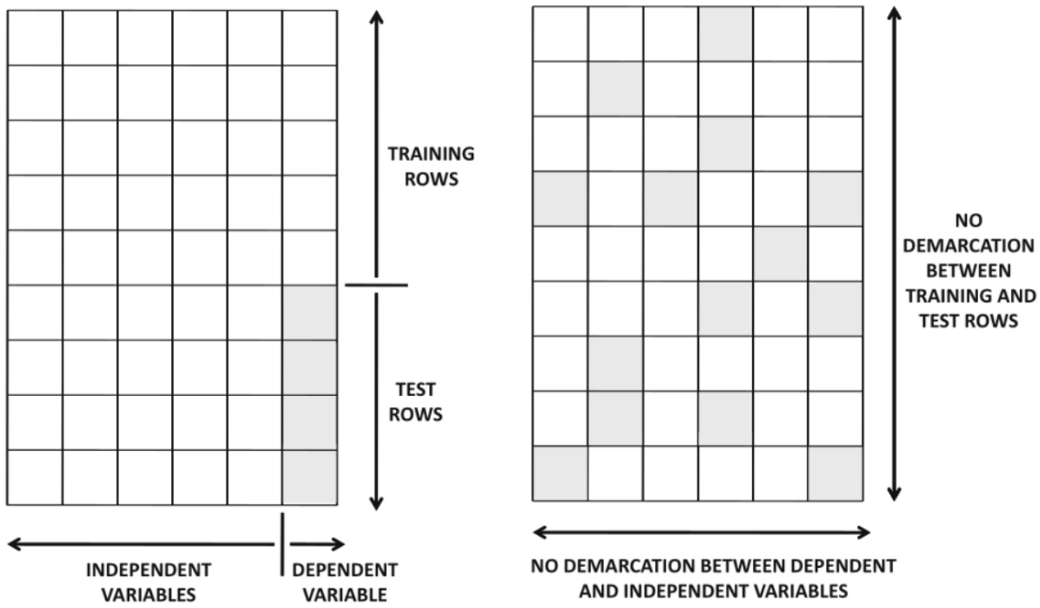
Система спочатку використовує вміст нового продукту для отримання рекомендацій, а потім, зрештою, дії користувача щодо цього продукту.

Методи, засновані на моделях, засновані на машинному навчанні та методах аналізу даних для прогнозування рейтингу користувачів без оцінки предметів.

Ці методи здатні рекомендувати більшу кількість елементів більшій кількості користувачів, порівняно з іншими методами, такими як заснована на пам'яті.

Приклади таких методів, що базуються на моделях, включають дерева рішень, моделі, що базуються на правилах, байєсівські методи та моделі прихованих факторів.

Відповідно, їх застосування застосовує параметри навчання, для чого необхідне розділення даних на навчальну та тестову вибірки, а також видання цільових та незалежних змінних (*див. Рис.2.12*).



(а) Колаборaційна фільтрація на основі моделі

(б) Колаборaційна фільтрація на основі пам’яті

Рисунок 2.12 – Розподіл даних при різних підходах

Методи колаборaційної фільтрації долають виклики, з якими системи рекомендації на основі вмісту не можуть впоратись. Вони мають можливість рекомендувати нові предмети, навіть якщо таких немає рейтинги, надані користувачами. Таким чином, навіть якщо база даних цього не робить містять налаштування користувача, точність рекомендацій – ні постраждалих.

Крім того, якщо налаштування користувача змінюються, він має можливість коригувати свої рекомендації за короткий проміжок часу. Вони може керувати ситуаціями, коли різні користувачі не діляться однакові предмети, але лише однакові предмети відповідно до їх власних властивостей особливості. Користувачі можуть отримувати рекомендації, не ділячись своїми профілю, і це забезпечує конфіденційність [15].

Техніка колабортивної фільтрації також може надати пояснення щодо формування рекомендацій для користувачів. Однак методики страждають від різних проблем, про що йдеться в літературі [12]. Фільтрування на основі вмісту методи залежать від метаданих елементів. Тобто вони вимагають розширеного опису предметів і дуже добре організованого користувача профілю, перш ніж рекомендація може бути надана користувачам. Це називається обмеженим аналізом вмісту.

Таким чином, ефективність колаборативної фільтрації залежить від наявності описових даних. Надмірна спеціалізація вмісту [20] - ще одна серйозна проблема техніки колаборативної фільтрації. Користувачі можуть отримувати рекомендації, подібні до елементів, уже визначених у їхніх профілях.

2.3.3 Гібридні рекомендатори (Hybrid recommenders)

Недавні дослідження показують, що для підвищення ефективності систем, що рекомендують, варто поєднувати спільні рекомендації та рекомендації на основі вмісту (*див. Рис.2.13*). Гібридні рекомендатори включають комбінацію колаборативної фільтрації та ситстем рекомендацій, побудованих на основі вмісту. Гібридні системи рекомендацій покращують прогнозування продуктивність, а також покращують складність та витрати на впровадження. Хоча, для цього потрібна зовнішня інформація бути функціональним, який в основному недоступний.

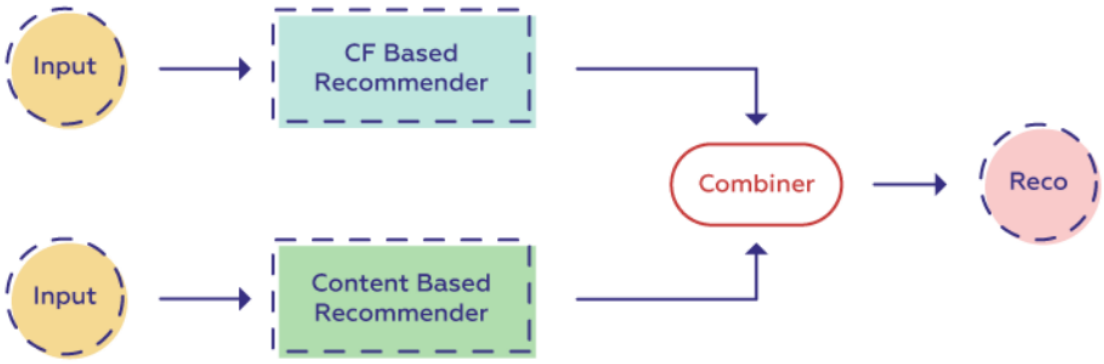


Рисунок 2.13 – Схема побудови гібридного рекомендатора

Гібридні підходи можуть бути реалізовані шляхом окремого прогнозування на основі вмісту та співпраці, а потім їх комбінування шляхом додавання можливостей на основі вмісту до підходу, що базується на співпраці, і навпаки; або шляхом об'єднання підходів в одну модель.

Наприклад, Netflix дає рекомендації, порівнюючи звички перегляду та пошуку схожих користувачів (тобто спільна фільтрація), а також пропонуючи фільми, що мають спільні характеристики, із фільмами, які користувач оцінив високо (фільтрація на основі вмісту).

2.4. Оцінка якості систем рекомендації

2.4.1. Оцінка за допомогою метрик точності

Якість рекомендаційного алгоритму можна оцінити за допомогою різних типів вимірювань, які можуть бути точністю чи охопленням. Тип використовуваних показників залежить від типу техніки фільтрування.

Точність - це частка правильних рекомендацій від загальної кількості можливих рекомендацій, тоді як охоплення вимірює частку об’єктів у просторі пошуку, для яких система може надати рекомендації. Метрики для вимірювання точності систем фільтрації рекомендацій поділяються на статистичні та метрики точності підтримки прийняття рішень [21]. Придатність кожної метрики залежить від особливостей набору даних та типу завдань, які буде виконувати система, що рекомендує [22].

Статистичні показники точності оцінюють точність техніки фільтрування шляхом порівняння прогнозованих оцінок безпосередньо з фактичним рейтингом користувача. Середня абсолютна похибка (MAE), середньоквадратична похибка (RMSE) та кореляція зазвичай використовуються як статистичні показники точності. MAE - найпопулярніший і найпоширеніший; це міра відхилення рекомендації від конкретного значення користувача. Він обчислюється наступним чином:

$$MAE = \frac{1}{N}\sum_{u,i}\left|p_{u,i} - r_{u,i}\right| \ ,$$

де $p_{u,i}$ - прогнозований рейтинг для користувача u по елементу i,

$r_{u,i}$ - фактичний рейтинг,

N - загальна кількість оцінок за набором елементів.

Чим нижче MAE, тим точніше механізм рекомендацій прогнозує рейтинги користувачів. Крім того, використовують також таку метрику як RMSE (Root Mean Square Error):

$$RMSE = \sqrt{\frac{1}{n}\sum_{u,i}(p_{u,i} - r_{u,i})^2}$$

RMSE робить більший акцент на більшій абсолютній похибці, і чим нижча RMSE, тим краща точність рекомендацій.

Популярними показниками точності підтримки прийняття рішень є Reversal rate, Weighted errors, Receiver Operating Characteristics (ROC) and Precision Recall Curve (PRC), Precision, Recall and F-measure. Ці показники допомагають користувачам вибирати елементи, які мають дуже високу якість із доступного набору елементів.

Метрики розглядають процедуру передбачення як двійкову операцію, яка відрізняє хороші предмети від тих, які не є хорошими. Криві ROC дуже успішні при виконанні комплексної оцінки продуктивності деяких конкретних алгоритмів. Точність - це частка рекомендованих елементів, яка насправді актуальна для користувача, тоді як відкликання може бути визначена як частка відповідних

елементів, яка також є частиною набору рекомендованих елементів [87]. Вони обчислюються як:

$$Precision = \frac{\text{Правильно рекомендовані елементи}}{\text{Загальна кількість рекомендованих елементів}}$$

$$Recall = \frac{\text{Правильно рекомендовані елементи}}{\text{Загальна кількість коричних рекомендованих елементів}}$$

F-міра, визначена нижче, допомагає спростити точність і відкликання в єдину метрику. Отримане значення робить порівняння між алгоритмами та між наборами даних дуже простим і зрозумілим [23]:

$$F - measure = \frac{2PR}{P + R}$$

Охоплення пов’язане з відсотком товарів та користувачів, яким система, що рекомендує, може надавати прогнози.

Прогнозування може бути практично неможливим, якщо жоден користувач або декілька користувачів не оцінили товар. Покриття можна зменшити, визначивши невеликі розміри району для користувача чи предметів [23].

2.4.2. Експертна оцінка

Для оцінки якості та побудованої системи рекомендації може також застосовуватись експертна оцінка. Експертне оцінювання є одним із способів отримання та використання знань фахівців про предметну область, в основу якого покладено ранжування. Ранжування — це процедура впорядкування будь-яких об’єктів за зростанням або спаданням деякої властивості.

Системи експертного оцінювання (СЕО) призначені для апріорного (до проведення дослідів) ранжування експертами факторів впливу (чинників) на певне явище. У контексті систем рекомендації, СЕО можуть використовуватись для оцінки якості системи рекомендації у порівнянні із системами-аналогами. Тоді експерти порівнюють декілька систем-аналогів, ранжуючи їх за допомогою своєї експертної оцінки від найбільш релевантного до найменш, замість факторів впливу в такому випадку обираються системи, які ми хочемо порівняти.

Розглянемо детально одну з найбільш популярних методик – одночасне ранжування. Спеціалістам пропонують ранжувати всі чинники, які включено в анкету, по ступеню їхнього впливу на певний показник. Найважливіший, з точки зору спеціаліста, чинник одержує ранг 1, менш суттєвий - ранг 2 і т.д. Якщо спеціаліст вважає два або більше чинників однаково важливими, він має право поставити їм однакові ранги. Розглянемо метод одночасного ранжування на прикладі.

Вісім експертів (NE=8) залучені до оцінки ступеня впливу дев'ятьох (К =9) вхідних змінних на вихідну змінну об’єкта керування. У табл.2.2, яка називається матрицею ранжування, наведено результати роботи фахівців. Треба визначити відносний ступінь впливу кожного з факторів на вихідну змінну об’єкта та перевірити узгодженість думок експертів.

Таблиця 2.2 – Матриця ранжування

Експерти	Вхідні змінні (фактори)								
	1	2	3	4	5	6	7	8	K = 9
1	1	7	2	3	6	4	5	8	8
2	2	3	6	4	9	1	8	7	5
3	1	4	3	2	6	1	5	1	2
4	3	6	5	4	7	1	6	5	2
5	2	5	3	4	7	6	6	1	2
6	1	5	5	2	6	6	7	3	4
7	2	5	6	3	7	8	9	1	4
NE = 8	1	3	4	2	5	1	6	1	2

Розв’язок. Оскільки в матриці ранжування є ранги, що співпали (так звані “зв’язані”), то приведемо її спочатку до нормального виду.

У нормальній матриці сума кожного рядка дорівнює K(K+1)/2. У нормальній матриці змінним, які в одному рядку мають однакові ранги, надають ранг, що дорівнює середньому значенню тих місць, які ці змінні поділили між собою.

Так, 8-й і 9-й вхідним змінним експертом 1 був наданий однаковий ранг – 8. У результаті приведенні матриці до нормального виду їм надають ранг (8+9)/2=8,5. Наведемо в табл.2.3 нормальну матрицю ранжування.

Таблиця 2.3 – Нормальна матриця ранжування

Експерти	Вхідні змінні										
	1	2	3	4	5	6	7	8	9	Повто-рення	T_i
1	1	7	2	3	6	4	5	8,5	8,5	2	6
2	2	3	6	4	9	1	8	7	5	–	–
3	2	7	6	4,5	9	2	8	2	4,5	3;2	30
4	3	7,5	5,5	4	9	1	7,5	5,5	2	2;2	12
5	2,5	6	4	5	9	7,5	7,5	1	2,5	2;2	12
6	1	5,5	5,5	2	7,5	7,5	9	3	4	2;2	12
7	2	5	6	3	7	8	9	1	4	–	–
8	2	6	7	4,5	8	2	9	2	4,5	3;2	30
$\sum_{i=1}^{NE} a_{ij}$	15,5	47	42	30	64,5	33	63	30	35		

Тепер за даними табл. 2.3 підрахуємо суми рангів $\sum_{i=1}^{NE} a_{ij}$, які набрав j-й фактор після опитування усіх NE експертів. Ці суми будуть основними показниками сили впливу факторів на досліджувану властивість.

З наведеного прикладу очевидно, що найбільший вплив на досліджувану вихідну змінну має 1-й фактор. Далі йдуть фактори 4, 8, 6, 9, 3, 2, 5, 7.

Після отримання результатів експертизи варто перевірити гіпотезу про наявність узгодженості у думках спеціалістів.

Перевірку виконують за допомогою коефіцієнта конкордації Кендалла Значення цього коефіцієнту знаходяться у діапазоні [0,1] . Чим краща узгодженість думок, тим більший W.

Статистичну значущість коефіцієнта W оцінюють шляхом перевірки статистичних гіпотез:

$$H_0: W = 0;$$
$$H_1: W \neq 0.$$

При підтвердженні основної гіпотези H_0 буде визнано, що думки експертів не узгоджені. При відхиленні цієї гіпотези, приймемо альтернативну гіпотезу H_1, що вказує на узгодженість думок. Критерієм перевірки гіпотези H_0 є критерій Пірсона (χ^2 - критерій). Для розрахунку W використовують дві формули:

- для незв'язаних рангів:

$$W = \frac{12\sum_{j=1}^{K}\Delta_j^2}{NE^2(K^3-K)}$$

- для зв'язаних рангів:

$$W = \frac{12\sum_{j=1}^{K}\Delta_j^2}{NE^2(K^3-K)-NE\sum_{j=1}^{NE}T_i},$$

де $\sum_{j=1}^{K}\Delta_j^2$ – сума квадратів відхилень суми рангів кожного фактору від загальної середньої суми рангів,

T_i – параметр, який враховує повторення рангів у відповідях експертів.

Зазначену суму квадратів обчислюють за формулою:

$$\sum_{j=1}^{K} \Delta_j^2 = \sum_{j}^{K} (\sum_{i=1}^{NE} a_{ij} - \frac{\sum_j^K \sum_{i=1}^{NE} a_{ij}}{K})^2$$

Вона характеризує розсіювання результатів ранжування.

Показник T_i розраховують для кожного і-го рядка (для кожного і-го експерта) таблиці ранжування (табл.2.3) за наступною формулою:

$$T_i = \sum_{l=1}^{L} (t_{il}^3 - t_{il}) ,$$

де L – кількість груп рангів, які повторюються у і-у рядку;

t_{il} – кількість повторень і-го рангу у відповідях і–го експерта.

Наприклад, у першого експерта двічі зустрічається тільки ранг 8,5 (L = 1), тому t_{il}=2, у 2-го експерта повторень немає (L = 0), у 3-го експерта тричі повторюється ранг 2 і двічі – 4,5, тому t_{31}=3; t_{32} = 2(L = 1).

Отже, для 1 – о експерта L =1; $T_i = 2^3 - 2 = 6$.

Для 3-го експерта L =2; $T_3 = 3^3 - 3 + 2^3 - 2 = 27 - 3 + 8 - 2 = 30$.

Значення критерію для зв'язаних і незв'язаних рангів розраховують за виразом:

$$\chi^2 = NE(K-1)W$$

Розраховане χ^2 порівнюють із табличним при обраному рівні значущості α і кількості степенів вільності NU = K – 1. Гіпотезу H_0 відкидають, а отже визнають узгодженість у думках спеціалістів тоді, коли виконується умова $\chi^2 \geq \chi^2_{tabl}$.

2.5. Висновки до розділу

У цьому розділі було досліджено етапи побудови систем рекомендації, розглянуто різні підходи та методи їх побудови. Також було проведено порівняльний аналіз цих методів, визначено переваги та недоліки кожного з них. Було досліджено поняття «холодного старту» та визначено, які методи з ними краще справляються. Окрім цього, у розділи було розглянуто методи та метрики оцінки якості рекомендаційних систем.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ РЕКОМЕНДАЦІЇ

При побудові системи рекомендації для формування інформаційної інфраструктури розумного міста необхідним є як дослідження предметної області, так і методології, що може застосовуватись для досягнення такої мети. Проте не менш важливим є формування власного бачення програми/додатку та визначення завдань, які повинна вирішувати конкретна система рекомендації, для того, щоб бути корисною користувачеві.

В рамках дослідження, що було проведено під час написання магістерської дисертації, було визначено, що система рекомендації буде реалізована у вигляді програми, що має на меті стати «помічником» користувачеві-жителю міста та допомагати йому у проведенні свого соціального та культурного життя. Система рекомендації повинна буде пропонувати цікаві для користувача пропозиції, такі як куди можна сходити на вихідних, який концерт відвідати або пропонувати користувачеві цікаві саме для нього новини тощо. Розроблення кожної системи рекомендації, як було уже зазначено у розділі 2, складається з трьох етапів:

1. Етап збору інформації.
2. Етап навчання системи рекомендації.
3. Етап рекомендації/прогнозу.

Розглянемо кожен із цих етапів більш детально на прикладі розроблення програми системи рекомендації для побудови інформаційної інфраструктури міста.

Етап збору інформації. На першому етапі дуже важливим є збір, пошук та обробка інформації, яка буде використовуватись для побудови системи рекомендації, оскільки якість зібраних даних напряму впливає на якість майбутньої системи рекомендації та на релевантність варіантів, що будуть рекомендуватися користувачеві.

У якості даних, які будуть основою для побудови системи рекомендації розумного міста було обрано датасет, що був побудований на основі опитування 1076 людей та опублікований PAVIC у січні 2017 року [26]. Це опитування було повністю анонімним і було спрямоване на покращення життя громадян у майбутньому розумному місті.

Ідея цього опитування полягає в отриманні точного розуміння реакції громадян на різні рекомендації у різних контекстах. Очевидно, респондентам було запропоновано вибрати серед набору з рекомендацій ті, які їх найбільше зацікавили б, якщо б вони були запропоновані у двох різних контекстах: у сонячний і теплий (20°C) весняний суботній день і в дощовий і холодний (8°C) суботній день взимку (іменований контекстом "Дощ").

Рекомендації стосувались різних тем: соціальних чи культурних заходів, знижок у ресторанах, корисної інформації про місто тощо, і людей запитували в кожному контексті, які вони хотіли б отримувати як push-повідомлення на свої телефони. Для кожного контексту респонденти могли дати кілька відповідей або взагалі не відповісти.

Датасет складається з трьох таблиць: (**items** – містить характеристику елементів, що можуть бути рекомендовані користувачеві; **ratings** – містить рейтинги, які користувачі надавали пропозиціям; **usersDescription** – містить описову характеристику користувачів).

У таблиці **items** *(див. Рис.3.1.)* представлено перелік елементів, які можуть бути рекомендовані системою рекомендацій у Smart City. Опис полів таблиці:

- перший стовпець – itemId;
- другий стовпець - опис товару. Н-д: «Пропозиція спортивного заходу";
- останні 8 стовпців представляють (як двійковий вектор) категорію, до якої належить елемент.

	itemId	itemDesc	item1	item2	item3	item4	item5	item6	item7	item8
0	0	Various sports activities proposal	1	0	0	0	0	0	0	0
1	1	The results of your favorite sports team	1	0	0	0	0	0	0	0
2	2	A concert of your favorite band	0	1	0	0	0	0	0	0
3	3	A concert of a symphony orchestra	0	1	0	0	0	0	0	0
4	4	A discount for your favorite clothes store	0	0	1	0	0	0	0	0

Рисунок 3.1 – Приклад записів таблиці items

У таблиці **ratings** *(див. Рис.3.2.)* представлено рейтинги користувачів рекомендованих елементів. Опис полів таблиці:

- перший стовпець - contextId. Наприклад: "1077";
- другий стовпець - itemId. Наприклад: "1";
- третя колонка - це рейтинг елемента ("0" – відхилено, "1" – прийнято);
- четвертий стовпець - userId. Наприклад: "1".

	contextId	itemId	rating	userId
0	1	1	1	1
1	1	2	1	1
2	1	3	0	1
3	1	4	0	1
4	1	5	0	1
...	...	...	...	...
38731	2152	14	0	1076
38732	2152	15	0	1076
38733	2152	16	0	1076
38734	2152	17	0	1076
38735	2152	18	0	1076

38736 rows × 4 columns

Рисунок 3.2 – Таблиця ratings

У таблиці **usersDescription** *(див. Рис.3.3.)* представлено опис користувачів за допомогою вектору їх ознак. Розглянемо опис полів таблиці:

- перший стовпець – contextId (ID пропозиції);
- друга колонка – вік;
- наступні два виміри – це стать (1 – чоловік, 0 – жінка);
- наступні 13 вимірів - це соціально-професійна категорія (SPC), представлена у вигляді двійкового вектора;
- наступні 13 вимірів представляють спеціальність користувача за допомогою двійкового вектора;
- наступні 10 вимірів - це ознаки користувача, представлені у вигляді двійкового вектора;
- наступні 8 вимірів представляють найвищий ступінь користувача;
- наступні 2 виміри визначають погоду / сезон. Сонце "1; 0 і дощ" 0; 1 ";
- останній стовпець - це userId.

	contextId	age	man	woman	4	5	6	7	8	9	10	11	12	13	14	15	...	userId
0	1	34	1	0	0	0	1	0	0	0	0	0	0	0	0	0	...	1
1	2	32	1	0	0	0	1	0	0	0	0	0	0	0	0	0	...	2
2	3	43	1	0	0	0	1	0	0	0	0	0	0	0	0	0	...	3
3	4	32	1	0	0	0	1	0	0	0	0	0	0	0	0	0	...	4
4	5	49	1	0	0	0	1	0	0	0	0	0	0	0	0	0	...	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2147	2148	18	1	0	1	0	0	0	0	0	0	0	0	0	0	0	...	1072
2148	2149	21	0	1	1	0	0	0	0	0	0	0	0	0	0	0	...	1073
2149	2150	22	0	1	1	0	0	0	0	0	0	0	0	0	0	0	...	1074
2150	2151	18	0	1	1	0	0	0	0	0	0	0	0	0	0	0	...	1075
2151	2152	18	1	0	1	0	0	0	0	0	0	0	0	0	0	0	...	1076

2152 rows × 51 columns

Рисунок 3.3 – Таблиця usersDescription

Таким чином маємо датасет, що складається з трьох таблиць та містить інформацію про певну характеристику користувачів та їх вподобання, а також перелік об’єктів, що можуть бути рекомендовані користувачеві-жителю міста.

При побудові системи буде враховуватись специфіка датасету та використовуватися наявна вхідна інформація про користувача. Так, у вхідних даних є досить обширна характеристика користувачів та їх оцінки тим об’єктам, що можуть рекомендуватись, проте немає розгорнутого опису самих об’єктів.

Таким чином, це буде враховуватись при побудові системи рекомендації та для розробки програми будуть використовуватись користувацько-орієнтовані методи побудови систем рекомендацій, що базуються здебільшого на характеристиках про користувача та їх вподобаннях, а не на характеристиках самих об’єктів рекомендації.

3.1. Фільтрація на основі користувачів

Як один із підходів побудови рекомендаційної системи було використано фільтрацію на основі подібності характеристик користувачів. Даний метод був обраний, виходячи зі специфіки датасету. При побудові рекомендаційної системи як основну використаємо таблицю usersDescription, у якій є велика кількість ознак, що містять інофрмацію про характеристику користувачів (такі як вік, стать, соціально-професійні характеристики і т.д.)

Першим кроком побудови системи рекомендації на основі цього підходу є побудова матриці подібності користувачів за двома метриками подібності:

1. Подібність за косинусом.
2. Евклідова відстань.

Варто зауважити, що при виконанні наступних етапів побудови персоналізованої рекомендації, потрібно уважно враховувати специфіку цих метрик, оскільки вони по-різному оцінюють подібності між векторами.

Результати розрахунку подібності за косинусом будуть знаходитись у проміжку [0, 1], де 0 – найменш подібні елементи, а 1 – найбільш подібні елементи. В той час результати розрахунку Евклідової відстані будуть знаходитись у проміжку від [0, ∞], де 0 – ідентичні елементи, а ∞ - найбільш неподібні елементи. Тобто при виборі користувачів, що подібні до вхідного користувача при подібності за косинусом обиратимемо найбільші значення (найбільш близькі до одиниці), а за метрикою Евклідової відстані – користувачів, шо отримали найменший показник.

Таким чином, за допомогою описаних вище двох метрик, будуємо матриці подібності користувачів. Розглянемо візуалізацію матриці подібності користувачів за косинусом на прикладі перших 100 записів та перших 100 колонок матриці *(див. Рис. 3.4.)*

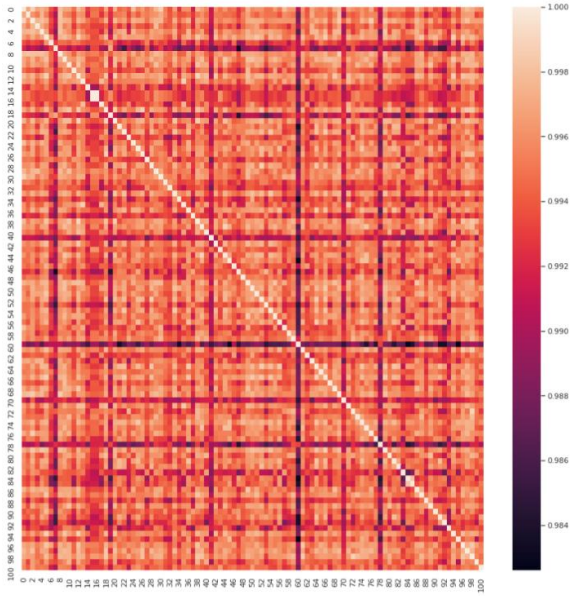


Рисунок 3.4 – Візуалізація матриці подібності користувачів за косинусом

Розглянемо візуалізацію матриці за Евклідовою відстанню на прикладі перших 100 записів та перших 100 колонок матриці (*див. Рис. 3.5.)*

Рисунок 3.5 – Візуалізація матриці подібності за Евклідовою відстанню

Таким чином, бачимо як попарно побідні користувачі за різними метриками на основі характеристик, що були надані у таблиці usersDescription вхідного датасету.

Наступним кроком є на основі отриманої матриці подібності знайти користуваччів, що є найбільш подібними користувачами, а отже таких, для яких вподобання у елементах будуть релевантними і для користувача, якому буде пропонуватись рекомендація. Розглянемо наступні етапи рекомендації на прикладі користувача з user_id=184.

Для вхідного користувача з user_id=184 найбільш подібними за характеристиками за подібністю косинуса є наступні користувачі (*див. Рис. 3.6*)

	userid	cosine_sim_score
0	194	0.998783
1	304	0.998780
2	465	0.998780
3	277	0.998755
4	727	0.998750
5	282	0.997647
6	186	0.997561
7	350	0.997561
8	357	0.997561
9	401	0.997561

Рисунок 3.6 – Найбільш подібні користувачі
за метрикою подібності за косинусом

Найбільш подібними для вхідного користувача за характеристиками за Евклідовою відстанню є наступні користувачі (*див.Рис. 3.7)*

	userid	euclidean_sim_score
0	194	1.000000
1	304	1.000000
2	465	1.000000
3	186	1.414214
4	277	1.414214
5	343	1.414214
6	346	1.414214
7	350	1.414214
8	357	1.414214
9	362	1.414214

Рисунок 3.7 – Найбільш подібні користувачі
за метрикою Евклідової відстані

Фінальним кроком є на основі отриманої матриці подібності знайти елементи, що є найбільш актуальними для подібних користувачів, а отже і для користувача, якому буде пропонуватись рекомендація. Відбираємо елементи, що найчастіше зустрічаються серед подібних користувачів. Результати, отримані за використання метрики подібності за косинусом зображено на *Рис.3.8*.

itemId	contextId	userId
6	18	18
9	12	12
2	12	12
10	12	12
1	10	10

Рисунок 3.8 – Найбільш релевантні рекомендації за подібністю за косинусом

Результати, отримані за використання метрики Евклідової відстані зображено на *Рис.3.9*.

itemId	contextId	userId
6	15	15
9	11	11
11	11	11
10	11	11
4	11	11

Рисунок 3.9 – Найбільш релевантні рекомендації за Евклідовою відстанню

3.2. Колаборaційна фільтрація (користувацько-орієнтована)

При розробці програми будуть використовуватись методи колаборативної фільтрації, що базуються на основі пам’яті. Як було описано в другому розділі, існує два підходи побудови таких систем – підхід, що базується на принципі «користувач-користувач» та підхід, що базується на принципі «елемент-елемент». Для розробки системи рекомендації на основі вмісту було обрано перший підхід.

Першим кроком для виконання колаборативної фільтрації є формування матриці рейтингів, що будується на основі таблиці ratings, де колонками виступає перелік об’єктів, рядками – користувачі, а значеннями – оцінки користувача відповідним об’єктам. Розглянемо візуалізацію матриці рейтингів на прикладі перших 100 засписів матриці (*див. Рис. 3.10)*

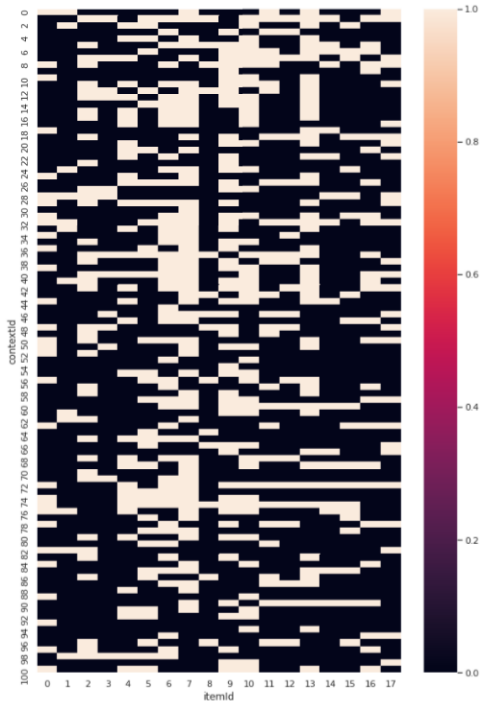


Рисунок 3.10 – Візуалізація матриці рейтингів

Наступні кроки будуть виконуватись уже персоналізовано для кожного користувача, якому ми захочемо рекомендувати ті чи інші елементи. Тому для наглядності наступні кроки будемо показувати на прикладі користувача з user_id=184.

Другим кроком є на основі отриманої матриці рейтингів знайти користувачів, що мають подібні вподобання до користувача, який ми обрали як того, кому ми будемо рекомендувати елементи.

Подібність користувачів знаходилась за допомогою розрахунки двох різних метрик подібності:

1. Подібність за косинусом.
2. Евклідова відстань.

Варто зауважити, що при виконанні наступних етапів побудови персоналізованої рекомендації, потрібно уважно враховувати специфіку цих метрик, оскільки вони по-різному оцінюють подібності між векторами.

Результати розрахунку подібності за косинусом будуть знаходитись у проміжку [0, 1], де 0 – найменш подібні елементи, а 1 – найбільш подібні елементи. В той час результати розрахунку Евклідової відстані будуть знаходитись у проміжку від [0, ∞], де 0 – ідентичні елементи, а ∞ - найбільш неподібні елементи.

Тобто при виборі користувачів, що подібні до вхідного користувача при подібності за косинусом обиратимемо найбільші значення (найбільш близькі до одиниці), а за метрикою Евклідової відстані – користувачів, шо отримали найменший показник. Таким чином, знаходимо попарно значення подібності вхідного користувача зі всіма іншими користувачами, використовуючи описані вище дві метрики.

Наступним кроком є ранжування користувачів за отриманими значеннями від найбільш подібного та найменш подібного та вибір перших топ N найбільш подібних користувачів.

Причому при використанні подібності за косинусом ранжування буде низхідним, а при використанні Евклідової відстані – висхідним. В результаті ранжування при використанні метрики подібності за косинусом та відбору перших топ 5 користувачів отримаємо *(див. Рис. 3.11)*:

	context_id	cosine_similirity
0	1025	0.800000
1	765	0.800000
2	1865	0.774597
3	962	0.774597
4	1978	0.745356
5	1610	0.707107

Рисунок 3.11 – Результати ранжування при подібності за косинусом

В результаті ранжування при використанні метрики Евклідової відстані та відбору перших топ 5 користувачів отримаємо *(див. Рис. 3.12)*:

	context_id	euclidean_distance
0	765	1.414214
1	962	1.414214
2	1025	1.414214
3	1865	1.414214
4	82	1.732051
5	154	1.732051

Рисунок 3.12 – Результати ранжування за Евклідовою відстанню

Бачимо, що при використанні обидвох метрик та ранжуванні значень серед перших топ 10 подібних користувачів є багато співпадінь, побудуємо спільну таблицю з результатами ранжування за обома метриками (*Рис. 3.13*)

	context_id	cosine_similirity	euclidean_distance
0	1025	0.800000	1.414214
1	765	0.800000	1.414214
2	1865	0.774597	1.414214
3	962	0.774597	1.414214
4	1978	0.745356	NaN
5	1610	0.707107	NaN
6	1126	0.707107	NaN
7	534	0.707107	NaN
8	342	0.707107	NaN
9	1617	0.676123	NaN
10	82	NaN	1.732051
11	154	NaN	1.732051
12	362	NaN	1.732051
13	374	NaN	1.732051
14	460	NaN	1.732051
15	627	NaN	1.732051

Рисунок 3.13 – Спільна таблиця результатів ранжування за двома метриками

Визначивши користувачів за метрикою подібності, необхідно наступним кроком визначити елементи, що будуть релевантними вхідному користувачу. Для цього обираємо користувачів, що були визначені як найбільш подібні та обираємо елементи, що відповідають їх вподобанням, за кожним елементом визначаємо середній рейтинг серед подібних користувачів та ранжуємо результати у низхідному порядку.

Результати ранжування (перші 10 записів) елементів за середнім рейтингом серед користувачів, що були визначені подібними до вхідного користувача за метрикою косинуса подібності зображено на *Рис. 3.14*.

contextId	1025	765	1865	962	1978	1610	1126	534	342	1617	mean_cosine
itemId											
0	1	1	1	1	1	1	1	1	1	1	1.0
4	1	1	1	1	1	1	1	1	1	0	0.9
1	0	1	1	1	1	1	1	1	1	1	0.9
16	1	1	0	0	1	1	1	1	1	1	0.8
17	1	0	0	0	1	1	1	1	1	1	0.7
10	0	0	0	0	1	1	1	0	1	1	0.5
7	0	0	0	0	1	0	1	1	1	0	0.4
15	0	0	0	0	0	1	0	1	1	1	0.4
14	0	0	0	0	0	1	0	1	1	0	0.3
2	1	0	0	0	0	0	1	0	0	1	0.3

Рисунок 3.14 – Найбільш релевантні елементи рекомендації за косинусом подібності

Результати ранжування (перші 10 записів) елементів за середнім рейтингом серед користувачів, що були визначені подібними до вхідного користувача за метрикою Евклідової відстані зображено на *Рис. 3.15*.

contextId	765	962	1025	1865	82	154	362	374	460	627	mean_euclidean
itemId											
0	1	1	1	1	1	1	1	1	0	1	0.9
4	1	1	1	1	0	1	0	1	0	0	0.6
1	1	1	0	1	1	0	0	0	0	1	0.5
17	0	0	1	0	1	1	1	0	1	0	0.5
16	1	0	1	0	0	0	0	0	1	0	0.3
2	0	0	1	0	1	0	0	0	0	0	0.2
5	0	0	0	0	0	1	0	0	0	0	0.1
6	1	0	0	0	0	0	0	0	0	0	0.1
3	0	0	0	0	0	0	0	0	0	0	0.0
7	0	0	0	0	0	0	0	0	0	0	0.0

Рисунок 3.15 – Найбільш релевантні елементи рекомендації за Евклідовою відстанню

Об’єднаємо результати рекомендації за обома метриками до однієї таблиці та оберемо перших топ 10 найбільш релевантних пропозицій (*Див. Рис. 3.16*)

	item_id	mean_cosine	mean_euclidean	mean
0	0	1.0	0.9	0.95
1	4	0.9	0.6	0.75
2	1	0.9	0.5	0.70
4	17	0.7	0.5	0.60
3	16	0.8	0.3	0.55
9	2	0.3	0.2	0.25
5	10	0.5	0.0	0.25
6	7	0.4	0.0	0.20
7	15	0.4	0.0	0.20
10	6	0.2	0.1	0.15

Рисунок 3.16 – Перші топ 10 найбільш релевантних пропозицій за обома метриками

Таким чином, ми отримали дві системи рекомендації, побудованих за різними підходами. Враховуючи специфіку кожної з них, можемо побудувати єдину систему, що буде використовувати переваги кожної із них на різних етапах взаємодії з користувачем. Оскільки фільтрація на основі користувачів будує рекомендації на основі характеристик про користувача, а не його історичних даних про попередні вподобання, цей підхід будемо застосовувати на перших етапах роботи з новими користувачами, що допоможе нам подолати проблему «холодного старту» та при цьому надавати релевантні рекомендації.

Важливим у роботі системи є зворотній зв’язок, після кожної рекомендації, будемо запитувати користувача, чи скористався він пропозицією та зберігати цю інформацію як історичні дані. Таким чином, коли набереться достатня кількість інформації про рекомендації зі зворотнім зв’язком (для роботи системи оберемо 50 записів), можемо переходити до другого підходу побудови та надання рекомендацій.

3.3. Оцінка якості системи рекомендації

Для оцінки якості роботи системи рекомендації порівняно з іншими системами-аналогами, проведемо дослідження з використанням експертної оцінки. Для початку визначимо системи-аналоги, з якими будемо порівнювати нашу систему. Оскільки повних аналогів, що надавали би саме персональні рекомендації щодо організації соціально-культурного життя немає, то візьмемо як аналоги сервіси, що надають пропозиції подій, заходів та тощо, хоч і не надають персоналізованих пропозицій. Такими сервісами є:

1. Веб-сервіс KudaGo

2. Телеграм-канал KudaKyiv

3. Мобільний додаток CityFrog

Для відбору експертів було обрано фокус-групу із 20 осіб, які повинні були протягом 20 днів користуватися розробленою системою рекомендації, а також системами-аналогами. Після чого серед учасників було проведено опитування на платформі Google Форми щодо користування цими додатками.

Ціллю було визначити, наскільки системи-аналоги є корисними для користувача порівняно з розробленою системою рекомендації. Для цього в анкеті було запропоновано оцінити кожен із додатку рангом від 1 до 10, де 1 – система надавала найбільш цікаві та релевантні пропозиції та 0 – система була нецікавою/неактуальною.

Також, для того, щоб визначити найбільш активних користувачів, крім проставленого рангу для кожної системи, було задано додаткові запитання: «Скільки пропозицій вам здалися цікавими у цьому додатку?» та «Скількома пропозиціями ви скористувались» у розрізі кожної із систем.

Таким чином, у результаті опитування було відібрано 10 найбільш активних користувачів, а на основі їх проставленого рангу для кожного з додатків, було побудовано експертну оцінку системи. Отримаємо матрицю ранжування (табл. 3.1).

Таблиця 3.1 – Матриця ранжування

Експерти	Системи			
	KudaGo	KudaKyiv	CityFrog	SmartCity
1	5	6	4	2
2	3	5	2	3
3	7	7	4	3

Продовження таблиці 3.1

Експерти	Системи			
	KudaGo	KudaKyiv	CityFrog	SmartCity
4	5	6	4	5
5	6	8	3	1
6	4	7	5	3
7	7	5	5	4
8	6	4	5	2
9	1	3	6	1
10	2	4	3	1

Оскільки в матриці ранжування є ранги, що співпали (так звані “зв’язані”), то приведемо її спочатку до нормального виду, так щоб сума кожного рядка дорівнювала би K(K+1)/2.

Таблиця 3.2 – Нормальна матриця ранжування

Експерти	Системи					
	KudaGo	KudaKyiv	CityFrog	SmartCity	Повто-рення	T_i
1	5	6	4	2	–	6
2	3	5	2,5	2,5	2	–
3	6,5	6,5	4	3	2	10
4	5	6	4	5	–	10
5	6	8	3	1	–	8

Продовження таблиці 3.2.

6	4	7	5	3	–	10
7	7	4,5	4,5	4	2	–
8	6	4	5	2	–	9
9	1,5	3	6	1,5	2	9
10	2	4	3	1	–	20
$\sum_{i=1}^{NE} a_{ij}$	46	54	41	25		

Тепер за даними таблиці 3.2 підрахуємо суми рангів $\sum_{i=1}^{NE} a_{ij}$, які набрав j-й фактор після опитування усіх NE експертів. Ці суми будуть основними показниками сили впливу факторів на досліджувану властивість.

З наведеного прикладу очевидно, що найкращою з точки зору експертної оцінки є побудована в магістерській дисертації система (Smat City), далі йдуть системи CityFrog, KudaGo, і KudaKyiv за експертною оцінкою є найгіршою із запропонованих варіантів.

Після отримання результатів експертизи варто перевірити гіпотезу про наявність узгодженості у думках спеціалістів.

Перевірку виконують за допомогою коефіцієнта конкордації Кендалла Значення цього коефіцієнту знаходяться у діапазоні [0,1] . За формулою, що детально описана у *розділі 2.4.2,* визначаємо, що коефіцієнт конкордації становить 0,83, а отже експерти узгоджені.

3.4. Висновки до розділу.

У розділі було розглянуто побудову системи рекомендації для вирішення задачі побудови додатку, що буде допомагати користувачам в організації свого соціально-культурного життя. Було розглянуто два підходи – побудова системи рекомендації, використовуючи особисті характеристики користувача (користувацько-орієнтоване фільтрування) та побудова системи рекомендації на основі його рейтингів та рейтингів інших користувачів (колаборaційна фільтрація). Причому варто зауважити, що перший підхід має наукову новизну, оскільки серед популярних існуючих методів немає таких, що враховували би при прогнозуванні персоналізованих рекомендацій описові характеристики користувача.

Також в результаті розробки системи рекомендації було розглянуто переваги та недоліки кожного з використаних підходів. Обидва методи дають релевантні рекомендації, проте краще проявлятимуть себе у різних ситуаціях. Метод фільтрації на основі характеристик про користувача є більш корисним при старті роботи з новим користувачам та допоможе боротися з «холодним стартом», в той час коли метод колаборативної фільтрації є актуальним, коли ми уже маємо багато інформації про переваги користувача, якому надається рекомендація та переваги подібних йому користувачів.

РОЗДІЛ 4. РОЗРОБЛЕННЯ СТАРТАПУ ПРОЕКТУ

4.1. Ідея проекту

Інформаційна карта, команда, опис ідеї та технологічна здійсненність стартап-проекту описані у табл. 4.1 - 4.4.

Таблиця 4.1 – Інформаційна карта проекту

1. Назва проекту	Smart city
2. Автори проекту	Мозолевська М.О.
3. Коротка анотація	Додаток буде надсилати персоналізовані рекомендації, сформовані за допомогою рекомендаційної системи. Користувач буде отримувати push-повідомлення із рекомендацією, а також може залишити фідбек щодо якості цієї рекомендації
4. Термін реалізації проекту	18 місяців
5. Необхідні ресурси	TyanGT20B7002 — 1 шт ~ 700$, процессор Intel «XeonE5620» — 2 шт ~ 760$, жёсткие диски WD «VelociRaptorWD1500HLFS» — 4 шт ~ 480$ 2 GbECCKingston — 8 шт ~ 540$ 200000$ витрати на персонал 20000$ витрати на приміщення

Продовження таблиці 4.1

6. Головні цілі та завдання проекту	Створення десктопного додатку для особистого користування та окремого продукту для промислового встановлення. Привернення уваги технологічних корпорацій до нашої команди для інвестицій у розробці більш важливих проектів, потребуючих великого фінансування.
7. Очікувані результати	Привернення уваги технологічних корпорацій до нашої команди для інвестицій у розробки більш важливих проектів, потребуючих великого фінансування, збір інформації

4.2 Команда стартапу

Таблиця 4.2 – Команда стартапу

Керівник проекту	Пошук інвесторів, керування компанією
Технічній директор	Керування технічною стороною проекту, розробка архітектури, підбір необхідних технологій. Досвід у програмуванні та менеджменті.
Програміст (2)	Розробка додатку. Досвід у машинному навчанні. Досвід створення серверних додатків.
Маркетолог	Розробка стратегій зацікавлення потенційних користувачів.Досвід популярізації продукту.
Веб-розробник	Створення сайту-платформи для взаємодії з серверним додатком. Навички у дизайні та веб-розробці.

4.3. Маркетингова стратегія та маркетинговий план стартапу.

4.3.1 Опис ідеї продукту

Таблиця 4.3 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Додаток, в основі якого за допомогою рекомендаційної системи користувач зможе вдосконалити своє соціально-культурне життя в місті	1. Використання персоналізованих рекомендацій	Користувач, за допомогою додатку може полегшити свій вибір, скориставшись персоналізованою рекомендацією
	2. Залишення оцінок та відгуків	Додаток надає можливість користувачеві залишити фідбек щодо місця/культурного заходу, порекомендованого системою. Таким чином система може надалі краще давати рекомендації як користувачеві, який залишив фідбек, так і іншим користувачам

Таблиця 4.4 – Технологічна здійсненність ідеї проекту

	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	ML	Caffe	Є у наявності	Відкрите програмне забезпечення
2	ML	Torch7	Є у наявності	Відкрите програмне забезпечення
3	ML	Tensorflow	Є у наявності	Відкрите програмне забезпечення
4	Database	MongoDB	Є у наявності	Відкрите програмне забезпечення
5	Database	PostgreSQL	Є у наявності	Відкрите програмне забезпечення
Обрана технологія Caffe + MongoDB				

За результатами аналізу таблиці можна зробити висновок щодо можливості технологічної реалізації проекту. У таблицях вище вказано, яке технологічне оснащення для цього необхідно, з поміж вказаних технологій відображено такі, що доступні авторам проекту та є наявними на ринку.

4.3.2 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Спочатку проводиться аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (табл. 4.5).

Рентабельність — поняття, що характеризує економічну ефективність виробництва, за якої за рахунок грошової виручки від реалізації продукції (робіт, послуг) повністю відшкодовує витрати на її виробництво й одержується прибуток як головне джерело розширеного відтворення.

Таблиця 4.5 – Попередня характеристика потенційного ринку стартап-проекту

	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	7
2	Загальний обсяг продаж, грн/ум.од	5$/ум.од
3	Динаміка ринку (якісна оцінка)	Стабільна
4	Наявність обмежень для входу (вказати характер обмежень)	Немає

Суть одного із найважливіших методів оцінки економічної ефективності інвестицій полягає у розрахунку їх середньої рентабельності за формулою:

$$R = P/\ 1{*}n{*}\ 100\ ,$$

де Р – прибуток за час експлуатації проекту; / - повна сума інвестиційних витрат; п - час експлуатації проекту.

Інвестувати грошові засоби доцільно тоді, коли від цього можна отримати більший прибуток, ніж від їх зберігання. Порівнюючи середньорічну рентабельність інвестицій зі ставкою банківського відсотка, можна дійти висновку, що вигідніше.

Середня норма рентабельності в галузі (або по ринку) порівнюється із банківським відсотком на вкладення. За умови, що останній є вищим, можливо, має сенс вкласти кошти в інший проект.

За результатами аналізу таблиці робиться висновок щодо, що ринок привабливим для входження за попереднім оцінюванням. Надалі проводиться аналіз ринкового середовища: складаються таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (табл. 4.6- 4.9). Фактори в таблиці подаються в порядку зменшення значущості.

Таблиця 4.6 – Фактори можливостей

Фактор	Зміст загрози	Можлива реакція компанії
Конкуренція	Збільшення зацікавленності великих гравців в цій галузі	Розробка унікального продукту, який буде більш якісним та мати цікаві нововведення

Продовження таблиці 4.6

Зміна потреб користувачів	Зміна пріорітетів користувачів, у зв’язку з якими, програмне забезпечення перестає задовольняти клієнта	Розширення функціональних можливостей додатку.
Низькі продажі	Додаток не затребуваний	Збільшити трати на рекламу
Низький ріст продажів	Погано поширюється	Розробити вірусну рекламу
Увага компаній	Привернення увагивеликих компаній до нашої команди	Розширення команди та початокроботи над дорогими проектами
Висока популярність додатку	Увага аудиторії до продукту дозволяє пропонувати їм нові продукти	Пропозиції укласти контракту із зацікавленними у розширенні аудиторії за рахунок реклами в нашому додатку

Надалі проведемо аналіз пропозиції: визначаються загальні риси конкуренції на ринку (табл. 4.7)

Таблиця 4.7 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - монополія/олігополія/ монополістична/чиста	Чиста	Ніякого. Клієнти отримують додаток безкоштовно - компанія отримує прибуток з реклами. Клієнти які купують додаток приносять прямий прибуток.
2. За рівнем конкурентної Боротьби - локальний/національний	Світовий	Сама природа мобільного додатку дозволяє пропонувати продукт всьому світові без затрат
3. За галузевою ознакою - міжгалузева/ внутрішньогалузева	Внутрішньо-галузева	Велика кількість інших розроблених додатків в яких наш може просто загубитися, тому необхідно розробити якір для уваги — слоган, назву, лого
4. Конкуренція за видами товарів: - товарно-видова - між бажаннями	Між бажаннями	Вкладення в різні види реклами, щоб переконати клієнта, що йому потрібен наш продукт

Продовження таблиці 4.7

5. За характером конкурентних переваг - цінова / нецінова	Нецінова	Конкурентні переваги – функції та можливості додатку
6. За інтенсивністю - марочна/не марочна	Марочна	Велика кількість інших розроблених додатків в яких наш може просто загубитися, тому необхідно розробити якір для уваги – слоган, назву, логотип додатку

За результатами аналізу таблиці робиться висновок щодо принципової можливості роботи на ринку з огляду на конкурентну ситуацію. Також робиться висновок щодо характеристик (сильних сторін), які повинен мати проект, щоб бути конкурентоспроможним на ринку. На основі аналізу конкуренції, проведеного в (табл. 4.7), а також із урахуванням характеристик ідеї проекту (табл. 4.3), та факторів маркетингового середовища визначимо та обґрунтовуємо перелік факторів конкурентоспроможності. Аналіз оформлюємо за табл. 4.8

Таблиця 4.8 – Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
Якість продукту	Один із факторів для вибору продукту клієнтом.

Продовження таблиця 4.8

Кількість видів соціальної взаємодії	Більша зацікавленість клієнта продуктом.
Ціна	Один із факторів для вибору продукту клієнтом.
Простота експлуатації	Один із факторів для зберігання зацікавленності клієнта.

За визначеними факторами конкурентоспроможності (табл. 4.8) проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 4.9).

Таблиця 4.9 – Порівняльний аналіз сильних та слабких сторін Smart-city

	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні						
			-3	-2	-1	0	1	2	3
1	Якість продукту	15			+				
2	Різноманіття функціоналу	10					+		
3	Ціна	10		+					
4.	Простота експлуатації	15			+				

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (табл. 4.10) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (табл. 4.9).

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими 75 результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення.

Наприклад: зниження доходів потенційних споживачів – фактор загрози, на основі якого можна зробити прогноз щодо посилення значущості цінового фактору при виборі товару та відповідно, – цінової конкуренції (а це вже – ринкова загроза).

Таблиця 4.10 – SWOT- аналіз стартап-проекту

Сильні сторони: ціна, простота використання	Слабкі сторони: досі невідома компанія
Можливості: наявність безкоштовного функціоналу	Загрози: видавлення з ринку конкурентами, не розуміння користувачем переваг

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок.

Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів (табл. 4.11).

З означених альтернатив обирається та, для якої:

1) отримання ресурсів є більш простим та ймовірним;
2) строки реалізації – більш стислими.

Таблиця 4.11 – Альтернативи ринкового впровадження стартап-проекту

Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
PR, просування бренду	50%	5
Перехід на безкоштовне розповсюдження	65%	3
Партнерство для об’єднання продукції	75%	3

4.4 Розроблення ринкової стратегії продукту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів (табл. 4.12)

Таблиця 4.12 – Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Поодинокі користувачі	Висока	Середній	Невисока	Низька
2	Малі компанії, зацікавлені в аналізі аудиторії людей.	Висока	Високий	Невисока	Середня
3	Крупні компанії,	Невисока	Невисокий	Висока	Середня
Які цільові групи обрано: 1,2					

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку:

- якщо компанія зосереджується на одному сегменті – вона обирає стратегію концентрованого маркетингу;

- якщо працює із кількома сегментами, розробляючи для них окремо програми ринкового впливу – вона використовує стратегію диференційованого маркетингу;

- якщо компанія працює із всім ринком, пропонуючи стандартизовану програму (включно із характеристиками товару/послуги) – вона використовує масовий маркетинг.

Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку (табл. 4.13).

Стратегія диференціації передбачає надання товару важливих з точки зору споживача відмітних властивостей, які роблять товар відмінним від товарів конкурентів. Така відмінність може базуватися на об’єктивних або суб’єктивних, відчутних і невідчутних властивостях товару(у ширшому розумінні – комплексі маркетингу), бути реальною або уявною. Інструментом реалізації стратегії диференціації є ринкове позиціонування.

Реалізація цієї стратегії вимагає, як правило, більш високих витрат. Проте успішна диференціація дозволяє компанії домогтись більшої рентабельності за рахунок того, що ринок готовий прийняти більш високу ціну (цінову премію бренду).

При веденні конкурентної боротьби з використанням цієї стратегії на ринку в першу чергу терплять фіаско фірми, що не здатні визначати потреби цільових ринків, оперативно реагувати на зміни в ринковому попиті, проводити ефективну політику маркетингових комунікацій, не мають необхідних навичок в області

брендингу. Найважливішими здібностями, які повинна мати компанія, що приймає цю стратегію, є з генерування маркетингових ноу-хау, здійснення продуктових новацій.

4.5. Розроблення маркетингової програми стартап-проекту

Таблиця 4.13 – Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання товару важливих з точки зору споживача відмітних властивостей	Стратегія диференціації	Більша кількість способів соціальної взаємодії, можливість переходу до безкоштовного типу розповсюдження	Стратегія диференціації

Наступним кроком є вибір стратегії конкурентної поведінки.

Компанії, що приймають слідування за лідером – це підприємства з невеликою часткою ринку, які вибирають адаптивну лінію поведінки на ринку, усвідомлюють своє місце на нім і йдуть у фарватері фірм-лідерів. Головна перевага такої стратегії – економія фінансових ресурсів, пов’язаних з необхідністю

розширення товарного(галузевого) ринку, постійними інноваціями, витратами на утримання домінуючого положення

Стратегія наслідування лідеру найчастіше має місце у випадку олігополії, коли кожен конкурент прагне уникнути боротьби, особливо цінової, а також у випадку, коли слабо виражений ефект масштабу, що не дозволяє отримати переваги від об'ємів продажів або ж він не грає істотної ролі. Стратегію наслідування лідеру приймають також фірми, які не змогли реалізувати стратегію виклику лідерові.

Компанії, що приймають таку стратегію, зазвичай випускають товари-імітатори, займаючи ринкову частку, яку з різних причин не можуть охопити фірми лідери. Вибір такої стратегії може також бути обумовлений також перевагою локалізації (краще знання ринку, налагоджені зв'язки з клієнтами тощо).

Для ефективної реалізації цієї стратегії компанії повинні задовольняти наступним основним умовам:

- систематичний аналіз сегментації ринку з метою виділення нових ринкових сегментів або таких, що незадовільно обслуговуються;
- ефективне використання НДДКР з метою вдосконалення технологічних процесів і незначних продуктових новацій;
- концентрація на прибутковості, а не на простому зростанні об'ємів продажів;
- залишатися досить малим, щоб не бути досить цікавим для фірм-лідерів;
- сильний керівник, здатний не лише формулювати стратегію, але і тримати усю діяльність компанії під власним контролем.
- Якщо врахувати, що лідерами ринку можуть бути лише декілька компаній, то ця стратегія наймасовішою.

Таблиця 4.14 – Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохід-цем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
Ні	Компанія буде як конкурувати за увагу існуючих споживачів, так і розширювати кількість споживачів за рахунок залучення нових.	Ні, функціонал значно відрізняється від функціоналу компаній-конкурентів за рахунок технологій, що дають можливість, на відміну від конкурентів, давати персоналізовані пропозиції	Експлерентна (піонерна) стратегія.

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки розробляється стратегія позиціонування (табл. 4.15), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Таблиця 4.15 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
Якість	Позиціювання за показниками якості	Тестування розробленого продукту та виправлення всіх багів	Стабільність роботи, якість роботи
Більша кількість функціональних можливостей	На основі специфічних відчутних характеристик	Розробка більшої кількості оригінальних можливостей	Великі можливості

Результатом виконання підрозділу має стати узгоджена система рішень щодо ринкової поведінки стартап-компанії, яка визначатиме напрями роботи стартап-компанії на ринку.

4.6. Висновки до розділу

В результаті переддипломної практики було досліджено предметну область майбутньої магістерської дисертації, а саме створення інформаційної інфраструктури міста за допомогою рекомендаційних систем, було досліджено поняття розумного міста та визначено, що основним напрямком розвитку буде покращення соціального та культурного життя мешканців міста.

Було досліджено методологію побудови рекомендаційних систем як інструменту, що може бути використаний для побудови інформаційної платформи розумного міста. Розглянуто типи систем рекомендації та дослідження досвід попередніх дослідників. У побудові системи рекомендації будуть використовуватись різні підходи та методологію та порівнюватись між собою.

Було проведено розбір проекту як можливого стартапу – в результаті дослідження було проаналізовано сильні та слабкі сторони проекту, маркетингові та фінансові можливості проекту тощо. Як результат можна зробити висновок, що проект має всі шанси на запуск.

ВИСНОВКИ

В умовах стрімкої урбанізації та розвитку інформаційних технологій все більш актуальним завданням стоїть побудова інформаційної інфраструктури міста, що допомагатиме його жителям в організації свого соціального та культурного життя. В наслідок цього важливою задачею є побудова систем, що, враховуючи персональні вподобання кожного користувача, надають рекомендації щодо різного роду активностей, яка може бути релевантна користувачеві.

Для реалізації цієї задачі хорошим інструментом є побудова систем рекомендацій. Під час виконання магістерської дисертації було розглянуто предметну область та досліджено підходи та методи, що використовуються для побудови рекомендаційної системи, що надає персоналізовані рекомендації для допомоги жителю міста в організації соціально-культурної сфери свого життя.

Як результат дослідження методів побудови рекомендаційних систем було перейнято як існуючі підходи їх побудови, так і внесено наукову новизну в прогнозування релевантності персональної пропозиції. Таким чином, було розроблено програму, яка надає користувачеві-жителю рекомендації, базуючись як на знанні про особисті характеристики користувача, так і на його вподобання.

ПЕРЕЛІК ПОСИЛАНЬ

1. Recommendation Systems – How Companies are Making Money. URL: https://sigmoidal.io/recommender-systems-recommendation-engine/
2. Badii C et.al. Analysis and assessment of a knowledge based smart city architecture providing service APIs. URL: https://www.sciencedirect.com/science/article/pii/S0167739X17302273
3. Аналіз великих даних для розумного міста за допомогою хмар. URL: https://doi.org/10.1109/UCC.2013.77
4. Moreno-Cano, V. et al. Big Data for IoT Services in Smart Cities. *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. Milan, Italy. 2015. P. 418-423.
5. Gaur, A., Scotney, B., Parr, G., McClean, S. Smart City Architecture and its Applications Based on IoT. *Procedia Computer Science*.2015.№52, P.1089–1094.
6. Anthopoulos LG, Vakali A. Urban planning and smart cities: interrelations and reciprocities. *The future internet FIA2012*. Berlin. 2012 P. 137.
7. Розуміння розумних міст: інтеграційна структура. URL: https://doi.org/10.1109/HICSS .2012.615.
8. Девід Б, Інь С, Чжоу Й та ін. SMART-CITY: Проблематика, методи та тематичні дослідження. *8-а міжнародна конференція з обчислювальних технологій та управління інформацією.* 2018. С. 168–174.
9. Nam T. Розумсне місто як міська інновація: зосередження уваги на управлінні, політиці та контексті. *5-а міжнародна конференція з теорії та практики електронного врядування*, 2011. С.185–194.
10. Su K. Розумне місто та програми. *Міжнародна конференція з електроніки, зв'язку та управління (ICECC)*, 2011. С. 1028–1031

11. Linden G., Smith B., and York J. Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, Los Alamitos, CA, USA. 2003. P. 76-80
12. Снитюк В. Е. Прогнозирование. Модели, методы, алгоритмы: учебное пособие. Київ: Маклаут, 2008. 364 с.
13. Dietmar Jannach, Markus Zanker, Alexander Felfernig, Gerhard Friedrich. Recommender Systems An Introduction. NY, USA : Cambridge University Press, 2011. 335 p.
14. J. Masthoff. Group recommender systems: Combining individual models. Recommender Systems Handbook. 2011. URL: https://www.researchgate.net/publication/227132202_Group_Recommender_Systems_Combining_Individual_Models
15. M. Mandl, A. Felfernig, E. Teppan, and M. Schubert. Consumer Decision Making in Knowledge-based Recommendation. *Journal of Intelligent Information Systems (JIIS)*. 2010. Vol. 18, No. 2, P. 37
16. Resnick, P., Varian, H.R.: Recommender systems. *Community of ACM 40.* 1997. Vol. 7, No 4, P. 56-58
17. J. Masthoff. Group recommender systems: Combining individual models. Recommender Systems Handbook. 2011. URL: https://www.researchgate.net/publication/227132202_Group_Recommender_Systems_Combining_Individual_Models
18. G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*. 2005. Vol. 41, No 5, P.734
19. R. M. Bell and Y. Koren. Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights, *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*. 2007. P. 43-52

20. R. M. Bell, Y. Koren, and C. Volinsky. The BellKor solution to the Net Flix Prize. 2007. URL: https://www.semanticscholar.org/paper/The-BellKor-solution-to-the-Netflix-Prize-Bell-Koren/f4ebb542c752a0dc423f94fd121e2edb8f6275ba
21. J. Bennett and S. Lanning. The Net Flix Prize. *13th ACM Int. Conf. on Knowledge Discovery and Data Mining.* San Jose, CA, USA, 2007. P. 22
22. Burke, R.Hybrid Recommender Systems: Survey and Experiments. User Modelling Interaction. 2002. URL: http://dx.doi.org/10.1023/A:1021240730564
23. R. Burke. Constraint-based recommender systems: technologies and research issues. *10th International Conference on Electronic Commerce (ICEC '08)*. New York. 2008 P.10
24. Dietmar Jannach, Markus Zanker, Alexander Felfernig, Gerhard Friedrich. Recommender Systems An Introduction. NY, USA : Cambridge University Press, 2011. 335 p.
25. R. Burke, M. Ramezani. Matching Recommendation Technologies. 2011. URL: https://www.researchgate.net/publication/226308926_Matching_Recommendation_Technologies_and_Domains
26. Recommendation System for Angers Smart City. 2017. URL: https://www.kaggle.com/assopavic/recommendation-system-for-angers-smart-city

ДОДАТОК А
ЛІСТИНГ ПРОГРАМНОГО КОДУ

```
import math
import seaborn as sns
sns.set(style="whitegrid", color_codes=True)

import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import sklearn
from sklearn.metrics.pairwise import cosine_similarity, euclidean_distances
from scipy.stats import pearsonr
import operator

%matplotlib inline
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = [16, 12]

import chardet
# set seed for reproducibility
np.random.seed(0)

#connect to Google Drive
from google.colab import drive
drive.mount('/content/drive')
from subprocess import check_output
files_path = "/content/drive/My Drive/smart_city_dataset/data/"
```

```
class Data:
  def __init__(self, path, *args):
    self.path = path
    self.filenames = check_output(["ls", self.path]).decode("utf8").strip().split('\n')
    # print(self.filenames)

  def _read_file(self):
    pass
class Data_Items (Data):
  def _read_file(self):
    print(self.filenames[0])
    df_items = pd.read_csv(self.path + self.filenames[0], sep =';', header = None)
    df_items = df_items.rename(columns = {0:'itemId', 1:'itemDesc', 2:'item1',3:'item2',4:'item3',
                                          5:'item4',6:'item5',7:'item6',8:'item7',9:'item8'})
    df_items['itemId'] = [i-1 for i in df_items['itemId']]
    print(f'Items df shape: {df_items.shape}' )

    return df_items
class Data_Ratings (Data):
  def _read_file(self):
    print(self.filenames[1])
    df_ratings = pd.read_csv(self.path + self.filenames[1], sep =';', header = None)
    df_ratings = df_ratings.rename(columns = {0:'contextId', 1:'itemId', 2:'rating',3:'userId'})
    df_ratings['contextId'] = [i-1 for i in df_ratings['contextId']]
    df_ratings['itemId'] = [i-1 for i in df_ratings['itemId']]
    df_ratings['userId'] = [i-1 for i in df_ratings['userId']]
    print(f'Ratings df shape: {df_ratings.shape}')

    return df_ratings
class Data_Descriptions (Data):
  def _read_file(self, print_shape = True):
    print(self.filenames[2])
```

```
    df_descriptions = pd.read_csv(self.path + self.filenames[2], sep =';', header = None)
    df_descriptions = df_descriptions.rename(columns = {0:'contextId', 1:'age', 2:'man', 3:'woman', 50:'u
serId'})

    for i in range(3,16):
      df_descriptions = df_descriptions.rename(columns = {i+1:'SPC'+str(i-2)})
    for i in range(16,29):
       df_descriptions = df_descriptions.rename(columns = {i+1:'userSpecialty'+str(i-15)})
    for i in range(29,39):
       df_descriptions = df_descriptions.rename(columns = {i+1:'userPreference'+str(i-28)})
    for i in range(39, 47):
       df_descriptions = df_descriptions.rename(columns = {i+1:'userHighDegree'+str(i-38)})
    for i in range(47,49):
       df_descriptions = df_descriptions.rename(columns = {i+1:'weatherSeason'+str(i-46)})

    df_descriptions['contextId'] = [i-1 for i in df_descriptions['contextId']]
    df_descriptions['userId'] = [i-1 for i in df_descriptions['userId']]

    if print_shape:
      print(f'Descriptions df shape: {df_descriptions.shape}')

    return df_descriptions
class Data_Users (Data):
  def _read_file(self):
    data_descriptions = Data_Descriptions(files_path)
    df_descriptions = data_descriptions._read_file(print_shape = False)

    df_users = df_descriptions.drop_duplicates(subset=['userId'])
    # user_ids = df_users.iloc[:, 50]
    df_users = df_users.iloc[:, 1:48] #delete contextid and weather features
    print(f'Users df shape: {df_users.shape}')
```

```
    return df_users
data = Data(files_path)
data._read_file()
data_items = Data_Items(files_path)
df_items = data_items._read_file()
df_items
data_ratings = Data_Ratings(files_path)
df_ratings = data_ratings._read_file()
df_ratings.head(10)
data_descriptions = Data_Descriptions(files_path)
df_descriptions = data_descriptions._read_file()
df_descriptions.head(10)
data_users = Data_Users(files_path)
df_users = data_users._read_file()
df_users.head(10)
df_ratings.columns.tolist()
 cosine_sim = cosine_similarity(df_users, df_users)
plt.subplots(figsize=(15,15))
sns.heatmap(cosine_sim_df.loc[:100, :100])
euclidean_sim = euclidean_distances(df_users, df_users)
euclidean_sim = pd.DataFrame(euclidean_sim)
euclidean_sim
plt.subplots(figsize=(15,10))
sns.heatmap(euclidean_sim.loc[:100, :100])
def get_recommendations(user_number, sim_df, similarity = 'cosine'):

    # Get the pairwsie similarity scores of all users with that user
    sim_scores = list(enumerate(sim_df[user_number]))

    # Sort the users based on the similarity scores
    if similarity == 'cosine':
      sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
```

```
      columns =['userid', 'cosine_sim_score']
    elif similarity == 'euclidean':
      sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=False)
      columns =['userid', 'euclidean_sim_score']

    # Get the scores of the 10 most similar users
    sim_scores = sim_scores[1:11]
    # print(sim_scores)

    # Get the user ids
    users_rec = pd.DataFrame(sim_scores, columns = columns)

    return users_rec
users_rec_cos = get_recommendations(184, cosine_sim)
users_rec_cos
cos_users_items = df_ratings[df_ratings['userId'].isin(users_rec_cos['userid'].to_list())]
cos_users_items
cos_users_items_pos = cos_users_items[cos_users_items['rating'] == 1]
cos_users_items_pos
recommended_items_cos = cos_users_items_pos.set_index(["itemId", "rating"]).count(level="itemId").
sort_values(by='userId', ascending=False)
recommended_items_cos.head(10)
users_rec_eucl = get_recommendations(184, euclidean_sim, similarity='euclidean')
users_rec_eucl
eucl_users_items = df_ratings[df_ratings['userId'].isin(users_rec_eucl['userid'].to_list())]
eucl_users_items
eucl_users_items_pos = eucl_users_items[eucl_users_items['rating'] == 1]
eucl_users_items_pos
recommended_items_eucl = eucl_users_items_pos.set_index(["itemId", "rating"]).count(level="itemId
").sort_values(by='userId', ascending=False)
recommended_items_eucl.head(10)
class Rating_Matrix:
```

```
  def __init__(self, df_ratings, *args):
    self.df_ratings = df_ratings
    # self.index = self.df_ratings.columns.tolist()[3] # index for rating_matrix
    self.index = self.df_ratings.columns.tolist()[0] # index for rating_matrix
    self.columns = self.df_ratings.columns.tolist()[1] # columns for rating_matrix
    self.values = self.df_ratings.columns.tolist()[2] # values for rating_matrix
    # self.user_id = user_id

  def get_rating_matrix(self):
    rating_matrix = self.df_ratings.pivot_table(index=self.index, columns=self.columns, values=self.val
ues)
    # replace NaN values with 0
    rating_matrix = rating_matrix.fillna(0)
    # display the top few rows

    return rating_matrix

class Collaborative_Recommendator:
  def __init__(self, user_id, rating_matrix, similarity = 'cosine_similarity', *args):
    self.user_id = user_id
    self.rating_matrix = rating_matrix
    self.similarity = similarity
    # self.get_similar_users = get_similar_users



  def get_similar_users(self, top_similar_users=10):
    user = self.rating_matrix[self.rating_matrix.index == self.user_id]
    other_users = self.rating_matrix[self.rating_matrix.index != self.user_id]

    # if similarity == 'pearson_correlation':
    #   user_values = user.values.tolist()[0]
    #   other_users = dict(zip(other_users.index.tolist(), other_users.values.tolist()))
```

```
#   # similarities = [pearsonr(user_values, other_users[i]) for i in other_users]
#   similarities = pearsonr(user_values, other_users[0])
# else:

if self.similarity == 'cosine_similarity':
  similarities = cosine_similarity(user,other_users)[0].tolist()

  # create list of indices of these users
  indices = other_users.index.tolist()

  # create key/values pairs of user index and their similarity
  index_similarity = dict(zip(indices, similarities))

  # sort by similarity
  index_similarity_sorted = sorted(index_similarity.items(), key=operator.itemgetter(1))
  index_similarity_sorted.reverse()

  # grab top n users off the top
  similar_users = index_similarity_sorted[:top_similar_users]
  similar_users = pd.DataFrame(similar_users, columns=['context_id', 'cosine_similirity'])
  # similar_users = pd.DataFrame(similar_users, columns=['user_id', 'cosine_similirity'])

elif self.similarity == 'euclidean_distances':
  similarities = euclidean_distances(user,other_users)[0].tolist()

  # create list of indices of these users
  indices = other_users.index.tolist()

  # create key/values pairs of user index and their similarity
  index_similarity = dict(zip(indices, similarities))

  # sort by similarity
```

```
        index_similarity_sorted = sorted(index_similarity.items(), key=operator.itemgetter(1))

        # grab top n users off the top
        similar_users = index_similarity_sorted[:top_similar_users]
        similar_users = pd.DataFrame(similar_users, columns=['context_id', 'euclidean_distance'])
        # similar_users = pd.DataFrame(similar_users, columns=['user_id', 'euclidean_distance'])

      return similar_users

    def get_recommended_items(self, items=10):
      print(self.similarity)
      # similar_users = self.get_similar_users()['user_id'].to_list()
      similar_users = self.get_similar_users()['context_id'].to_list()

      # rating_matrix = self.rating_matrix
      # items_matrix = self.rating_matrix[self.rating_matrix['user_id'] in similar_users]

      return similar_users

rating = Rating_Matrix(df_ratings)
rating_matrix = rating.get_rating_matrix()
rating_matrix
##example for user_id = 184

current_user = 184
cosine_recomendator = Collaborative_Recommendator(current_user, rating_matrix)

# try it out
cosine_similar_users = cosine_recomendator.get_similar_users()
cosine_similar_users
```

```
euclidean_recomendator = Collaborative_Recommendator(current_user, rating_matrix, similarity = 'eu
clidean_distances')

euclidean_similar_users = euclidean_recomendator.get_similar_users()
euclidean_similar_users
cosine_similar_users_list = cosine_recomendator.get_recommended_items()
print(cosine_similar_users_list)

# rating_matrix[rating_matrix.index == similar_users_list]
personal_users_matrix_cosine = rating_matrix.loc[cosine_similar_users_list, :].transpose()
# personal_users_matrix = personal_users_matrix.transpose()
personal_users_matrix_cosine['mean_cosine'] = personal_users_matrix_cosine.mean(axis=1)
items_cosine = personal_users_matrix_cosine.sort_values(by='mean_cosine', ascending=False)
items_cosine['item_id'] = items_cosine.index
items_cosine
euclidean_similar_users_list = euclidean_recomendator.get_recommended_items()
print(euclidean_similar_users_list)

# rating_matrix[rating_matrix.index == similar_users_list]
personal_users_matrix_euclidean = rating_matrix.loc[euclidean_similar_users_list, :].transpose()
# personal_users_matrix = personal_users_matrix.transpose()
personal_users_matrix_euclidean['mean_euclidean'] = personal_users_matrix_euclidean.mean(axis=1)
items_euclidean = personal_users_matrix_euclidean.sort_values(by='mean_euclidean', ascending=Fals
e)
items_euclidean['item_id'] = items_euclidean.index
items_euclidean
items_cosine[['mean_cosine', 'item_id']]
items_cosine_euclidean = pd.merge(items_cosine[['item_id', 'mean_cosine']], items_euclidean[['item_i
d', 'mean_euclidean']], how='outer')
items_cosine_euclidean = items_cosine_euclidean.rename_axis(None, axis = 1)
items_cosine_euclidean['mean'] = (items_cosine_euclidean['mean_cosine'] + items_cosine_euclidean['
mean_euclidean'])/2
```

```
items_cosine_euclidean = items_cosine_euclidean.sort_values(by='mean', ascending=False)
items_cosine_euclidean
```